UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola d'Enginyeria de Barcelona Est

Treball Final de Grau

Creació d'una app mòbil pel control d'un terrari

Creation of a mobile app for the control of a terrarium

Enginyeria Electrónica, Industrial i Automàtica

Primavera 2018/2019

Autor: Rubén Navas Soler

Director: Antoni Perez-Poch

Departament: CS (Computer Science)

Índex General

Índex de Figures 6
Índex de les Taules 8
Agraïments 9
Resum 10
Resumen 11
Abstract 12
1.- Introducció 13
1.1.- Objectiu 14
1.2.-Estructura de la memòria 15
2.- Funcionament del Sistema 16
2.1.- Aplicació mòbil 17
2.2.- Comandament a distància 21
3.- Planificació 25
4.- Hardware 27
4.1.- Arduino 27
4.2.- Mòduls y sensors 31
4.3.- Altres elements 40
4.4.- Costos 43
5.- Pressupost i Impacte Ambiental 45
5.1.-Pressupost 45
5.2.-Estudi de mercat 47
5.3.-Impacte Ambiental 48
6.- Software 49
6.1.-Diagrama de blocs del sistema 56
7.- Explicació codi Android 57
7.1.-Portada 58
7.2.- DispositivosBT 62

7.3.-Principal 66
8.- Explicació codi Arduino 80
9.- Conclusions 91
9.1.-Futures línies de treball 92
Bibliografia 93
Annex I: Codi Android Studio 96
AndroidManifest.xml 96
Portada.java 97
DispositivosBT.java 99
Portada.java 101
FondoControl.xml 113
Menu_popup.xml 113
Humedad.xml 114
Temperatura.xml 115
activity_portada.xml 116
activity_dispositivos_bt.xml 117
activity_principal.xml 118
colors.xml 125
dimens.xml 136
styles.xml 136
strings.xml 137
Annex II: Arduino 138
IRremoteInt.h 150
SimpleDHT.h 160
LiquidCristal.h 161
AnnexIII: Datasheets 163
Arduino Mega 2560 163
AX-1838HS 168
DHT11 173
DS3231 181

HL-54S 185
HC-06 187
LCD1602A 192
Annex IV: Esquema de connexió 193

Índex de Figures

Figura 2.1.- Posició inicial del camaleó 17
Figura 2.2.- Posició intermèdia del camaleó 17
Figura 2.3.- Posició final del camaleó 17
Figura 2.4.- Menú emergent al polsar 18
Figura 2.5.-Pantalla de configuració 18
Figura 2.6.-Pantalla del panell de control. 19
Figura 2.7.- Comandament a distancia utilitzat. 21
Figura 2.8.-Pantalla inicial de la LCD. 22
Figura 2.9.-Pantalla al polsar EQ 22
Figura 2.10.-Pantalla al polsar 4. 22
Figura 2.11.-Pantalla al polsar 5 o 6.. 23
Figura 2.12.-Pantalla al no rebre bé el valor.. 23
Figura 2.13: Imatge del estat inicial del terrari 23
Figura 2.14.-Resultat final del sistema creat 24
Figura 3.1.-Diagrama de Gantt 25
Figura 4.1: Esquema del Arduino Mega 2560. 30
Figura 4.2: Imatge del Sensor DHT11. 31
Figura 4.3: Esquema de connexió del sensor DHT11 32
Figura 4.4:Imatge del mòdul HC-06.. 33
Figura 4.5:Imatge del mòdul ZS-040.. 33
Figura 4.6: Esquema de connexió del mòdul ZS-040.. 34
Figura 4.7: Imatge del mòdul HL-45S.. 35
Figura 4.8: Esquema de connexió del mòdul HL-45S.. 35
Figura 4.9: Imatge del mòdul d'infrarojos AX-1838HS. 36
Figura 4.10: Imatge de la pantalla LCD 1602A.. 37
Figura 4.11: Esquema de connexió de la pantalla LCD 1602A.. 38
Figura 4.12: Imatge del rellotge ZS-042.. 39

Figura 4.13: Símbols Electronics del dos tipus de transistor BJT 41
Figura 4.14:Esquema del ordre dels pins que formen el PN2222..... 41
Figura 4.15:Esquema electrònic de la resistència i del potenciòmetre.. 42
Figura 4.16: Imatge del ventilador. 42
Figura 4.17: Imatge del portalàmpades.. 43
Figura 6.1.- Logotip de l'Android Studio.. 49
Figura 6.2.- Carpetes de l'aplicació.. 50
Figura 6.3.- Carpeta manifests. 50
Figura 6.4.- Carpeta java. 51
Figura 6.5.- Carpeta res. 52
Figura 6.6.- Cicle de vida d'una activitat a Android Studio.. 53
Figura 6.7.- Logotip de l'Arduino IDE.. 54
Figura 6.8.- Exemple d'un programa nou d'Arduino.. 55
Figura 6.9.- Diagrama de blocs del sistema... 56
Figura 7.1.- Diagrama de blocs Android..... 57
Figura 7.2.- Diagrama de blocs de Portada. 58
Figura 7.3.- Diagrama de blocs de DispositivosBT. 62
Figura 7.4.- Diagrama de blocs de Principal..... 66
Figura 8.1.- Diagrama de l'arduino. 80

Índex de les Taules

Taula 4.1: Tipus de plaques Arduino segons el microcontrolador. 28
Taula 4.2: Codi de colors de l'esquema. 30
Taula 4.3: Característiques elèctriques del DHT11. 32
Taula 4.4: Característiques elèctriques del ZS-040. 33
Taula 4.5: Característiques elèctriques del HL-45S. 35
Taula 4.6: Característiques elèctriques del AX-1838HS. 37
Taula 4.7: Característiques elèctriques de la pantalla LCD 1602A. 37
Taula 4.8: Característiques elèctriques del rellotge ZS-042.. 39
Taula 4.9: Característiques elèctriques del PN2222 41
Taula 4.10: Costos inicials. 43
Taula 4.11: Costos finals. 44
Taula 5.1: Taula del preu d'hores treballades. 45
Taula 5.2: Taula del pressupost. 45
Taula 5.3: Taula del càlcul del cost per unitat. 46
Taula 5.4: Taula dels beneficis. 46
Taula 5.5: Taula del càlcul del cost per unitat final. 46

Agraïments

Primer de tot agrair als meus pares per tot el suport donat al llarg del grau cursat i per proporcionar-me el suport econòmic per dur a terme aquest projecte.

Agrair l'ajuda de tots els companys al llarg de tota la carrera, ja que sense el seu ajut hagués sigut més complicat.

Finalment agrair al meu tutor, l'Antoni, per donar-me l'oportunitat de participar en aquest projecte.

Resum

L'objectiu d'aquest projecte és la creació d'un sistema que sigui capaç de controlar les funcions bàsiques d'un terrari on habita un camaleó, mitjançant una placa de desenvolupament hardware, en aquest cas un Arduino, controlat per una aplicació de telèfon mòbil. Per arribar a l'objectiu, el telèfon es connecta via Bluetooth a un mòdul incorporat a la placa de desenvolupament. L'aplicació és capaç de: visualitzar la temperatura i humitat del terrari mitjançant un sensor, el control sobre tres portalàmpades, una de calor i dos d'il·luminació, el control de l'hora l'encesa/apagament d'aquests i de la posada en mode automàtic de la temperatura. A part, incorpora un comandament a distància per si es donés el cas de la impossibilitat d'utilitzar el telèfon. A més, el sistema incorpora una pantalla que mostra en tot moment la data actual i els valors de la temperatura i humitat, i mitjançant el comandament pot mostrar altres valors d'interès.

Resumen

El objetivo de este proyecto es la creación de un sistema capaz de controlar las funciones básicas de un terrario donde habita un camaleón, mediante una placa de desarrollo hardware, en este caso se usa un Arduino, controlado por una aplicación de teléfono móvil. Para llegar al objetivo, el teléfono se conecta vía Bluetooth a un módulo incorporado a la placa de desarrollo. La aplicación es capaz de: visualizar la temperatura y la humedad del terrario mediante un sensor, el control sobre los portalámparas, una de calor y dos de iluminación, el control de la hora de encendido/apagado de estas, y la puesta en modo automático de la temperatura. A parte, incorpora un mando a distancia, por si se da el caso de la imposibilidad del uso del teléfono móvil. Además, el sistema incorpora una pantalla que muestra en todo momento la fecha actual y los valores de temperatura y humedad, y mediante el mando puede mostrar otros valores de interés.

Abstract

The objective of this project is the creation of a system capable of controlling the basic functions of a terrarium inhabited by a chameleon, using a hardware development board, in this case an Arduino, controlled by a mobile phone application. To reach the objective, the phone connects by Bluetooth to a module built into the development board. The application is able to: visualize the temperature and humidity of the terrarium by a sensor, the control on the lampholders, one of heat and two of lighting, the control of the time of on / off of these, and the putting in automatic mode of the temperature. In addition, it incorporates a remote control, in case of the impossibility of using the mobile phone. Also, the system incorporates a screen that shows at all times the current date, temperature and humidity values, and through the command can show other values of interest.

1.- Introducció

El camaleó de Yemen (Chamaelo Calytraputus) és un rèptil arbori que quan habita en llibertat ho fa en els altiplans, boscos i a les valls de les zones muntanyoses. Els hi agrada viure a la cima dels arbres o arbustos prop del terra.

Aquesta espècie de rèptil es caracteritza per la seva agressivitat, i el seu caràcter territorial, ja que prefereixen viure en solitari. Cada camaleó és únic en el seu color i forma de la pell. És un animal capaç de canviar el color de la pell segons la situació emocional.

El "Calytraptus" en situació de tranquil·litat té un color verd clar amb unes línies grogues. En situació de perill l'animal adopta un color més fosc i l'hi apareixen uns talps negres per tota la pell i el seu cos s'infla. En el cas que estigui estressat, l'animal es posarà de color blanquinós. S'estressa amb els colors vermells i blancs. És un rèptil que no té orelles per tant no pot escolar però percep les vibracions mitjançant els ulls.

L'alimentació consta d'insectes, siguin grills, paneroles, cucs o mosques. En captivitat el camaleó és considerat un animal exòtic, el qual ha d'habitar dins d'un terrari rodejat de plantes (siguin vives o de plàstic) i pals o rames per tal que l'animal pugui escalar i moure's.

El terrari ha d'estar entre 25 i 30 °C durant el dia i superior a 18 °C durant la nit i ha de tenir una humitat relativa entre el 50-80%.

Aquest està equipat amb dos portalàmpades, una que proporciona calor i l'altre llum fluorescent que és indispensable pel camaleó, ja que la intensitat de la llum influeix en el comportament de caça i incrementa la seva vitalitat.[1][2]

1.1.- Objectiu

L'objectiu d'aquest projecte és el de crear un sistema capaç de fer el control automàtic de la il·luminació i de calor d'un terrari.

Aquest sistema de control és governat per una placa de desenvolupament hardware, un Arduino, el qual incorporarà una pantalla LCD per visualitzar hora i data actual, temperatura i humitat. Per l'adquisició d'aquests valors s'utilitza un rellotge i un sensor connectats al microcontrolador (Arduino). Amb la incorporació del rellotge es fa el control automàtic de la il·luminació, i amb el sensor de temperatura/humitat es fa el control apropiat de la calor.

Per encendre i apagar la part d'il·luminació s'incorporen uns relés, també a l'Arduino. Per tal de poder tindre domini sobre el microcontrolador es crea una aplicació mòbil pel sistema operatiu Android. L'aplicació és capaç de visualitzar temperatura i humitat, tenir control manual sobre l'encesa/apagada dels relés i de marcar les hores d'encesa i apagada automàtica de la il·luminació. També és capaç d'iniciar i parar el mode automàtic del control de la calor, que al terrari se simbolitza amb un ventilador . La comunicació entre microcontrolador i telèfon mòbil és via Bluetooth, per tant hi ha un mòdul connectat a l'Arduino per poder establir la comunicació.

En el cas que no es disposi del telèfon mòbil, està incorporat un sistema de control mitjançant un comandament a distància, que unit a un sensor d'infrarojos, són capaços de substituir, en gran mesura, a l'aplicació mòbil. Aquest comandament té control per mostrar valors d'interès de la pantalla LCD i apagar-la si fos necessari.

Aquest prototipus pot resoldre la problemàtica de què l'animal que viu dins del terrari, en aquest cas un camaleó, tingui una regularitat en les hores de llum i la quantitat de calor proporcionada i tenir informació dels valors de camp en el terrari.

1.2.-Estructura de la memòria

La memòria s'estructura en diverses parts. La primera és l'explicació del treball realitzat i la planificació d'aquest, una segona part que consta d'un resum de tots aquells elements usats per la creació del sistema i un pressupost del projecte. La tercera part està formada per un breu resum dels programes usats i l'explicació més extensa del codi creat. Finalment la memòria consta de la bibliografia i dels Annexos, en els quals s'incorporen els codis de programació complets, els Datasheets dels elements usats i l'esquema de connexió.

2.- Funcionament del Sistema

La finalitat del projecte és la de millorar el control del terrari d'un camaleó que inicialment només consta de dos portalàmpades, el primer incorpora una llum que proporciona calor i el segon proporciona llum de raig ultraviolats. El sistema creat consta de dues parts, l'aplicació mòbil i el microcontrolador.
L'aplicació mòbil és capaç de controlar l'estat del portalàmpades, encès o apagat, designar l'hora d'apagada de la llum principal i secundària, una vegada designada, automàticament es posarà l'hora d'apagat de la calor, que són dues hores més i l'hora d'encesa de totes tres, que serà al cap d'onze hores. En l'aplicació es mostrarà totes les hores, per defecte, en el cas que és reinici el sistema, l'hora d'apagada de les dues llums és a les 21:00, la calor a les 23:00 i l'hora d'encesa és a les 08:00. També és capaç de controlar el mode automàtic, que se simbolitza amb el ventilador de la caixa de connexions. En aquest mode s'activarà la llum secundària, en el cas que estigui desactivada, i desactivarà la calor en el moment que la temperatura sigui superior a 31 °C. Per altra banda, passarà el procés contrari en el cas que la temperatura sigui inferior a 28 °C. L'aplicació també mostra els valors de temperatura i humitat. L'ús de l'app de mòbil és diferent del quotidià, s'assembla més a un ús industrial.

Aquest tipus d'aplicacions en la indústria s'anomenen SCADA, el qual consisteix en una aplicació software capaç de comunicar-se amb els dispositius de camp (el microcontrolador i els seus mòduls) i té control en el procés (les funcions del terrari) des de la pantalla.

El microcontrolador és qui du a terme les accions demanades pel telèfon. En ell estan connectats els relés, el sensor de temperatura i humitat, la pantalla LCD, el mòdul Bluetooth i el sensor d'infrarojos. La pantalla mostra per defecte la temperatura i la humitat. Mitjançant un comandament a distància, permet la visualització de l'hora d'encesa de cada portalàmpades, l'hora d'apagada i també incorpora algunes funcions igual que l'aplicació mòbil.

Aquest comandament està principalment, per si es dóna el cas que no es disposi del telèfon mòbil. En aquest apartat s'explicarà més a fons el funcionament de l'aplicació mòbil i del comandament.

2.1.- Aplicació mòbil.

A l'iniciar l'app, sortirà una pantalla inicial que consta d'un camaleó animat el qual es va movent sobre una branca. En les següents imatges s'observa la transició.

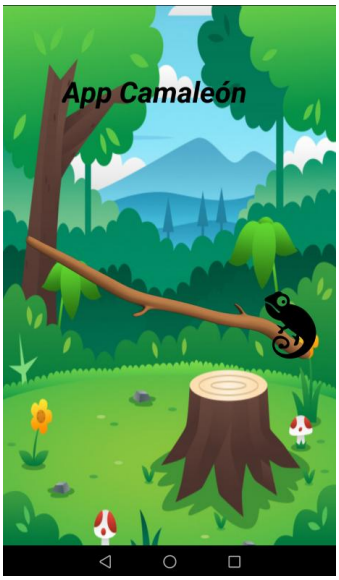


Figura 2.1.- Posició inicial del camaleó

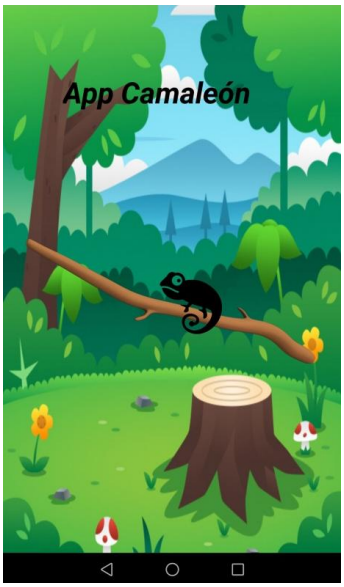


Figura 2.2.- Posició intermèdia del camaleó

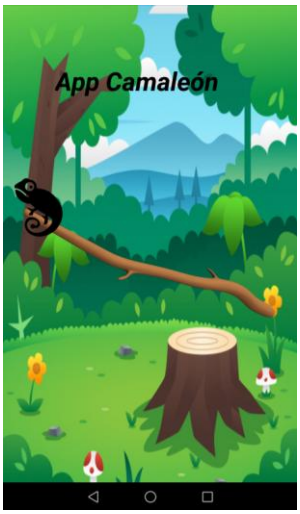


Figura2.3.- Posició final del camaleó

Per iniciar la comunicació Bluetooth amb la placa, s'ha de polsar al camaleó, en qualsevol moment del moviment, i sortirà un menú el qual permet iniciar o sortir de l'aplicació.

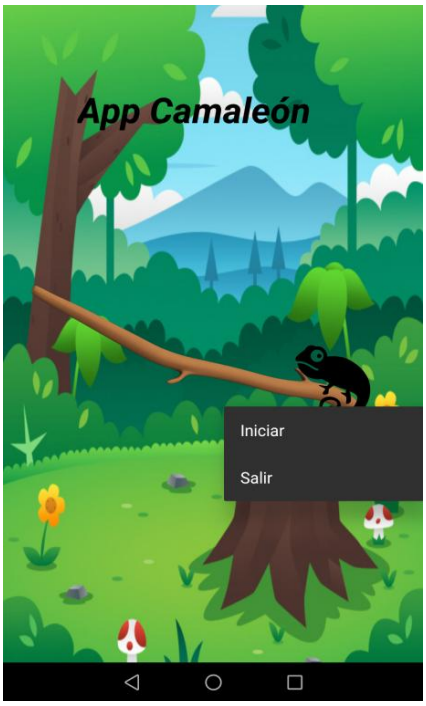


Figura 2.4.- Menú emergent al polsar el camaleó

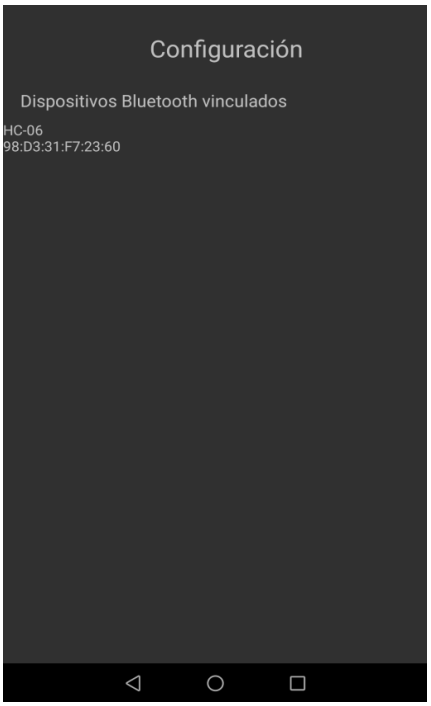


Figura 2.5.-Pantalla de configuració

Un cop polsat el botó iniciar, es canviarà de pantalla direcció a la configuració d'aparellament .

Dins d'aquesta pantalla apareixeran tots els dispositius Bluetooth vinculats amb el telèfon mòbil. En el cas que el Bluetooth estigui apagat, l'aplicació demana a l'usuari i al telèfon d'encendre-la.

En la figura 2.5 es pot observar que només hi ha un dispositiu vinculat al telèfon, aquest és el mòdul de Bluetooth connectat a la placa. Un cop polsat, si es pot crear la connexió entre mòbil i placa, s'avançarà a la següent pantalla, en el cas contrari hi ha l'opció que faci l'intent de canviar de pantalla i tornar a aquesta o que torni a la pantalla anterior.

En el cas que la connexió sigui creada, la següent pantalla és el panell de control.

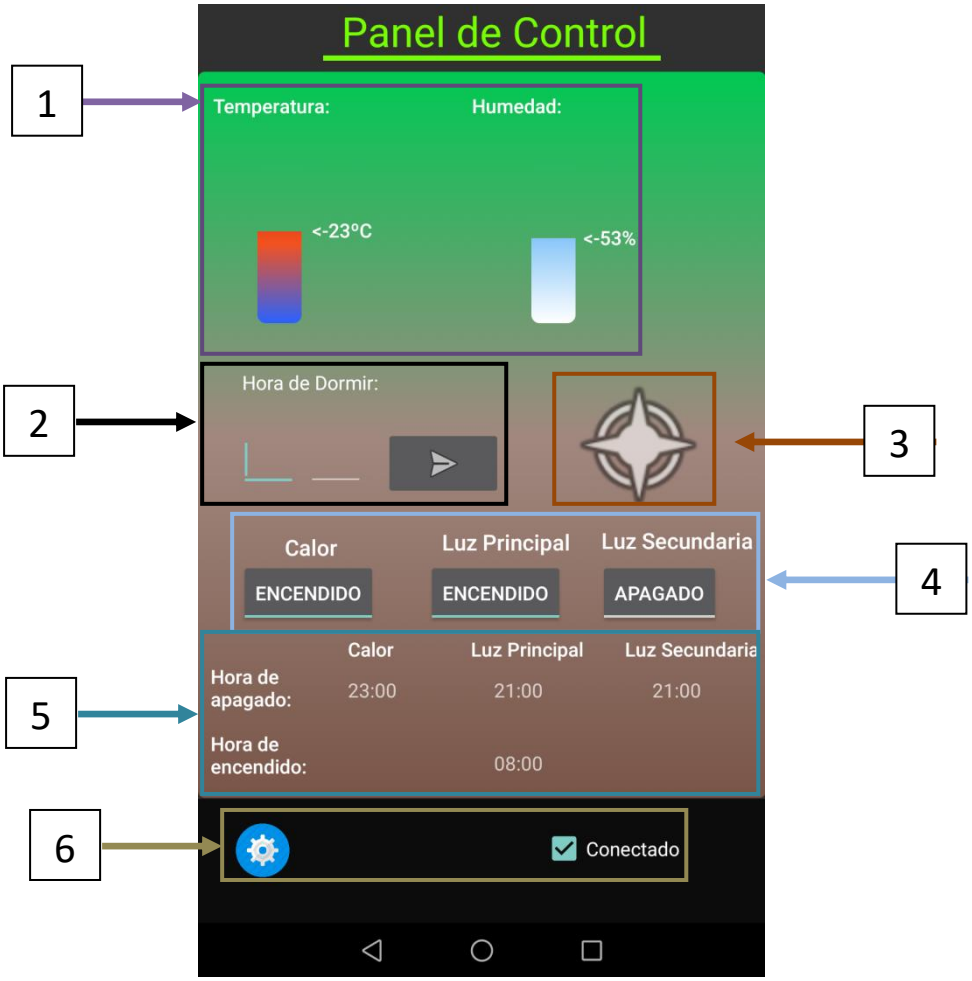


Figura 2.6.-Pantalla del panell de control.

1. Visualització de la temperatura i la humitat, depenent del valor, la barra augmentarà o disminuirà, i de la mateixa manera el valor estarà al mateix nivell de la barra.

2. Configuració de l'hora d'apagada de les llums.

3. Encesa o apagada del ventilador de la caixa de connexió, que també activa el mode automàtic, el qual està configurat de tal manera que si la temperatura augmenta a més de 30 graus, la calor s'apagaria i s'encendria la llum secundària i en el cas que la temperatura estigués per sota de 28 graus s'encendria la calor i s'apagaria la llum secundària. Per determinar que està activat, la imatge rota.

4. Els tres botons controlen manualment els tres portalàmpades i també es mostra l'estat d'ells. En el cas que el mode automàtic estigui encès, no hi ha control sobre la calor si la temperatura és menor a 28 ºC i sobre la llum secundària en el cas que la temperatura sigui major a 30 ºC.

5. Representació de les hores d'apagada de cada portalàmpades i l'hora de l'encesa, en totes és la mateixa. Les hores per defecte són les mostrades en la figura anterior.

6. Polsar l'engranatge comporta tornar a la pantalla anterior, és a dir, a la selecció del dispositiu Bluetooth. El "tic" mostra si el dispositiu està connectat, en cas contrari es mostraria el text "Desconectado" i si en el temps no hi ha connexió, es tornaria a la pantalla anterior.

Aquest és el funcionament de l'aplicació, l'explicació del codi està a l'apartat 7.

2.2.- Comandament a distància

Com a segona opció pel control de la placa s'ha optat per la utilització d'un comandament, ja que poden haver-hi situacions on no es pugui usar el telèfon mòbil i sigui necessari fer alguna acció sobre la placa.

El comandament té les mateixes funcions que el mòbil excepte que no es pot configurar l'hora d'apagada. Hi ha accions del comandament que tenen repercussió sobre la pantalla LCD.

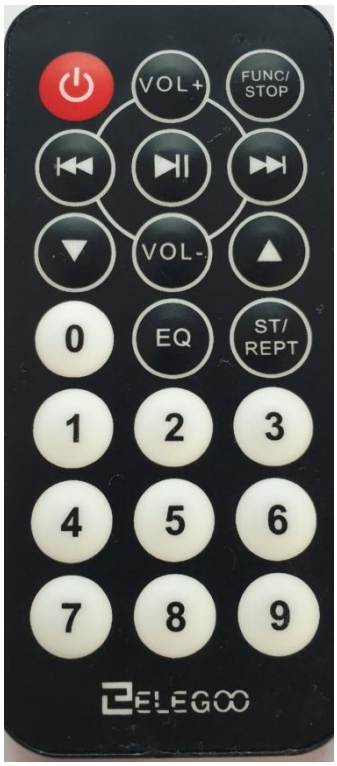


Figura 2.7.- Comandament a distància utilitzat.

Els botons i les seves funcions utilitzats són:

- Power: Encén i apaga la pantalla LCD.
- Play: Apaga i encén el ventilador/ mode automàtic.
- EQ: Mostra per la segona línia de la pantalla LCD l'hora d'encesa com es pot observar a la figura 2.9.
- 0: Reinicia tots els valors, tant les hores d'apagada, les d'encesa i els portalàmpades.

- 1: Apaga/Encén la calor en el cas que el mode automàtic estigui apagat.
- 2: Apaga/Encén la llum principal.
- 3: Apaga/Encén la llum secundària.
- 4: Mostra l'estat de la calor i l'hora d'apagada, figura 2.10.
- 5: Mostra l'estat de la llum principal i l'hora d'apagada, figura 2.11.
- 6: Mostra l'estat de la llum secundària i l'hora d'apagada, figura 2.11.

Aquest són els botons i les seves funcions, en el cas que no rebí bé el receptor, es mostrarà per pantalla "XXXX" entre el dia i l'hora, com es pot observar en la Figura 2.12.

Figura 2.8.-Pantalla inicial de la LCD.

La pantalla inicial de la LCD és la que es pot observar en la figura anterior. Es mostra el dia i hora actuals, la temperatura i la humitat.

L'única manera de variar el contingut de la pantalla és mitjançant el comandament.

A continuació es mostren canvis en la pantalla LCD:

Figura 2.9.-Pantalla al polsar EQ.

Figura 2.10.-Pantalla al polsar 4.

Figura 2.11.-Pantalla al polsar 5 o 6.

Figura 2.12.-Pantalla al no rebre bé el valor.

Aquest és el funcionament del sistema creat, el qual mitjançant una placa de desenvolupament hardware, Arduino en aquest cas, s'adquireix el control sobre la calor i la il·luminació. Aquest sistema proporciona autonomia a l'hora d'estar pendent del control de la son de l'animal, i de la calor, ja que s'obté un valor aproximat de la temperatura a l'interior del terrari.

Comparació del terrari

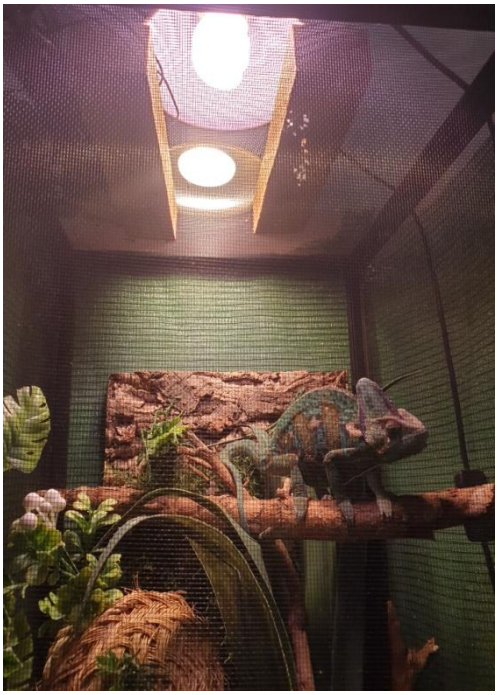


Figura 2.13: Imatge del estat inicial del terrari.

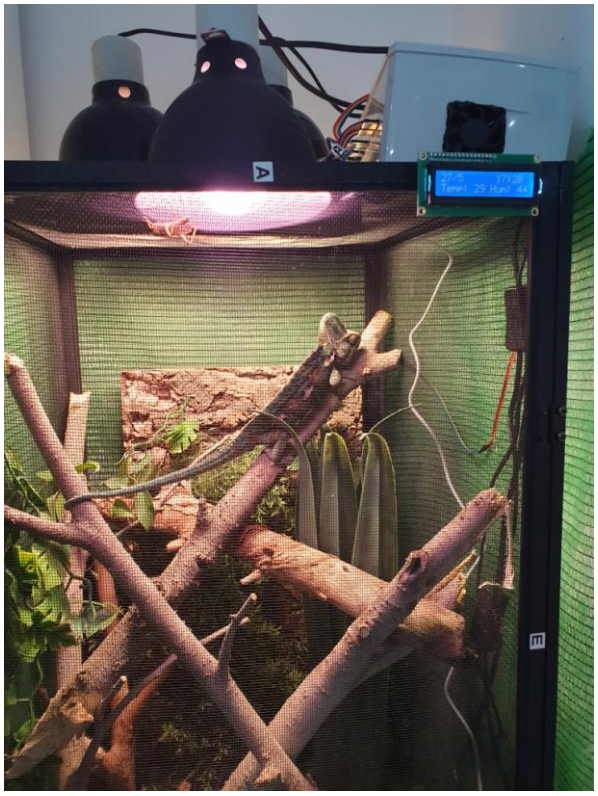


Figura 2.14.-Resultat final del sistema creat.

Per poder fer una comparació inicial i final, es pot observar la figura 2.13 que seria el sistema inicial i la figura 2.14 que correspondria al sistema creat i implementat al terrari. En aquesta última figura s'observa el tercer portalàmpades, la pantalla LCD i la caixa on està el microcontrolador i totes les connexions. Aquest sistema proporciona una estabilitat i una constància en el moment del control del horari de llum del camaleó, i amb l'incoporació de la llum secundaria dòna més il·luminitat al terrari.

3.- Planificació

En aquest apartat s'exposen les hores utilitzades per a la creació del projecte i l'explicació de les tasques dutes a terme.

La millor manera d'observar les hores invertides en les tasques del projecte és mitjançant un diagrama de Gantt.

ID	Nom de la tasca	Inici	Duración	Fi
1	Instalació i primer contacte Android Studio	2019-02-18	2.0 Semana	2019-03-05
2	Creació de l'entorn de l'app	2019-03-06	4.7 Semana	2019-04-12
3	Primer contacte Arduino	2019-03-06	1.5 Semana	2019-03-18
4	Programa arduino	2019-03-18	3.3 Semana	2019-04-12
5	Comunicació	2019-03-29	3.0 Semana	2019-04-22
6	Juntar App+Arduino	2019-04-13	3.0 Semana	2019-05-07
7	Implementació del sistema	2019-05-07	0.5 Semana	2019-05-10
8	Memòria	2019-05-01	4.1 Semana	2019-06-03

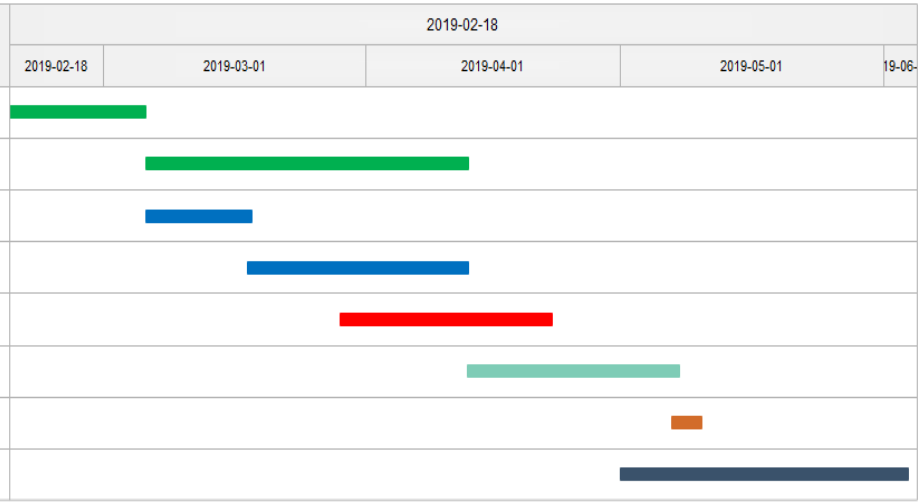


Figura 3.1.-Diagrama de Gantt.

La figura anterior correspon al diagrama de Gantt, on es pot observar que hi ha 8 tasques diferents:

1.- Instal·lació i primer contacte a Android Studio: Instal·lació del programa Android Studio i primer contacte amb el programa per a familiaritzar-se.

2.- Creació de l'entorn de l'app: Un cop iniciada la programació amb Android, es comença a fer l'entorn de l'aplicació. És la tasca amb més temps dedicat, ja que era una programació desconeguda.

3.- Primer contacte Arduino: Instal·lació de l'Android IDE i comprovació del funcionament dels mòduls adherits al microcontrolador.

4.- Programa Arduino: Un cop comprovat el funcionament, es crea el programa que enllaça els diversos mòduls i el correcte funcionament d'ells.

5.- Comunicació: Creació de la comunicació entre Arduino i Android.

6.- Ajuntar Aplicació i Arduino: Un cop establerta la comunicació entre ells, es crea el conjunt del sistema.

7.- Implementació del sistema: Un cop acabat totes les funcionalitats del sistema, s'instal·la al terrari.

8.-Memòria: Creació del document escrit a lliurar.

S'han invertit un total de 14 setmanes en la creació d'aquest projecte, i si a cada setmana s'ha treballat al voltant de 6 dies setmanals, surt un total de 84 dies, a 6-7 hores diàries suma un total de 504-588 hores invertides.

Punts Crítics:

A continuació s'exposen els punts més crítics, a nivell de temps dedicat, del projecte:

1.- *Comunicació Ethernet:* Durant el període de comunicació s'ha intentat establir la comunicació Ethernet però en sobrepassar el temps marcat per aquesta tasca i com que no s'ha arribat a l'objectiu d'establir una comunicació clara entre Arduino i telèfon mòbil es decideix prescindir de la comunicació Ethernet i quedar-se amb el Bluetooth.

2.- *Sensor de temperatura i d'infrarojos:* En incorporar el sensor d'infrarojos provocava que el sensor de temperatura no funciones correctament. Després de dies invertits, es troba el problema què en incorporar la llibreria del sensor d'infrarojos es crea una interrupció i això provoca que el sensor de temperatura no llegeixi els valors. Incorporant una funció que para la interrupció a l'hora de llegir se soluciona aquest problema.

3.- *Càmera:* Durant el període del programa de l'Arduino s'ha aconseguit l'obtenció d'imatges mitjançant una càmera com un mòdul de l'Arduino però la incorporació de la càmera fa més lent el sistema. I si això li sumes que la comunicació és mitjançant el Bluetooth no té sentit la incorporació d'aquesta, ja que el rang de la comunicació és reduït, per tant, a l'estar connectat al sistema, s'estarà a prop.

4.- Hardware

En aquest apartat s'explica tot el que envolta la part constructiva del projecte, la part física. Aquests elements són l'Arduino, els mòduls utilitzats i altres elements que donen sentit a les connexions. També es calculen els costos d'aquests elements.

L'esquema de connexions complet està inclòs a l'Annex IV.

4.1.- Arduino

Arduino és una placa de circuit imprès simple, basada en el microcontrolador de codi obert provinent de la plataforma de codi obert Wiring amb l'objectiu de fer més simple i accessible el disseny de circuits electrònics amb microcontroladors.

Consisteix en dissenys simples de maquinari lliure amb processadors Atmel AVR en una placa amb pins Entrades/Sortides.

L'entorn de desenvolupament implementa el llenguatge Processing de Wiring (el qual serà explicat en l'apartat de software), molt semblant a Java.

Arduino es pot utilitzar per desenvolupar objectes interactius autònoms o pot ser connectat a programari de l'ordinador.[3]

Tipus de plaques:

Existeixen diferents versions d'Arduino. En funció del projecte podem triar la que s'adapti millor. Una classificació de les plaques ve determinada pel tipus de connexió a l'ordinador:

- Placa sèrie:

És la placa bàsica, i s'utilitza una interfície RS232. Aquesta interfície pot ser utilitzada, a més, per a la programació de la placa, per a comunicar-se amb altres elements externs que utilitzin el port sèrie, com per exemple un PC.

- Placa USB:

És una evolució de la placa sèrie que incorpora un port USB per a comunicar-se amb el PC.

- Placa de prototips:

Aquesta placa està pensada per a poder incorporar hardware addicional al disseny base de l'Arduino. Incorpora una matriu de forats en la que s'hi pot afegir el nostre hardware addicional. No disposa de port sèrie ni USB, per la qual cosa és necessari disposar d'una altra placa per a programar el xip.

- Bluetooth:

És l'última versió en la qual es treballa. Elimina la necessitat de cables per a comunicar-se amb el PC o qualsevol altre dispositiu Bluetooth, com per exemple un telèfon mòbil.

En aquest projecte el tipus de placa que s'utilitza és una placa USB.

Una altra classificació de les plaques USB és segons el tipus de microcontrolador Atmel que incorpora.

A continuació es mostrarà els tipus que n'hi ha i algunes especificacions són:

Arduino	Processador	Flash	EEPROM	SRAM	PINS E/S	PWM	Pins Analògics
Diecimila	ATmega168	16	0.5	1	14	6	6
Due	Atmel 91SAM3X8E	512	NO	96	54	12	12
Duemilanove	ATmega168	16/32	0.5/1	1/2	14	6	6
Fio	ATmega328	32	1	2	14	6	8
LilyPad	ATmega168 or ATmega328	16	0.5	1	14	6	6
Mega	ATmega1280	128	4	8	54	14	16
Mega 2560	ATmega2560	256	4	8	54	14	16
Micro	ATmega32u4	32	1	2.5	20	7	7
Nano	ATmega168 or ATmega328	16/32	0.5/1	1/2	14	6	8
Uno	ATmega328P	32	1	2	14	6	6

Taula 4.1: Tipus de plaques Arduino segons el microcontrolador.

- Processador: Part de la placa que interpreta les instruccions contingudes en els programes i processa les dades.

- Memòria Flash: Medi d'emmagatzematge no volàtil que està desenvolupada a partir de la memòria EEPROM.

- Memòria EEPROM: És un tipus de memòria ROM que pot ser programada, esborrada i reprogramada elèctricament. Són memòries no volàtils.

- Memòria SRAM: Tipus de memòria d'ordinador primària que normalment s'utilitza en la implementació de la memòria cau dels ordinadors.

- PWM: Tipus de sortides de la placa que tenen la capacitat de generar sortides analògiques des de pins digitals.

En aquest projecte el tipus de placa USB que s'utilitza és l'Arduino Mega 2560 el qual té l'esquema següent:

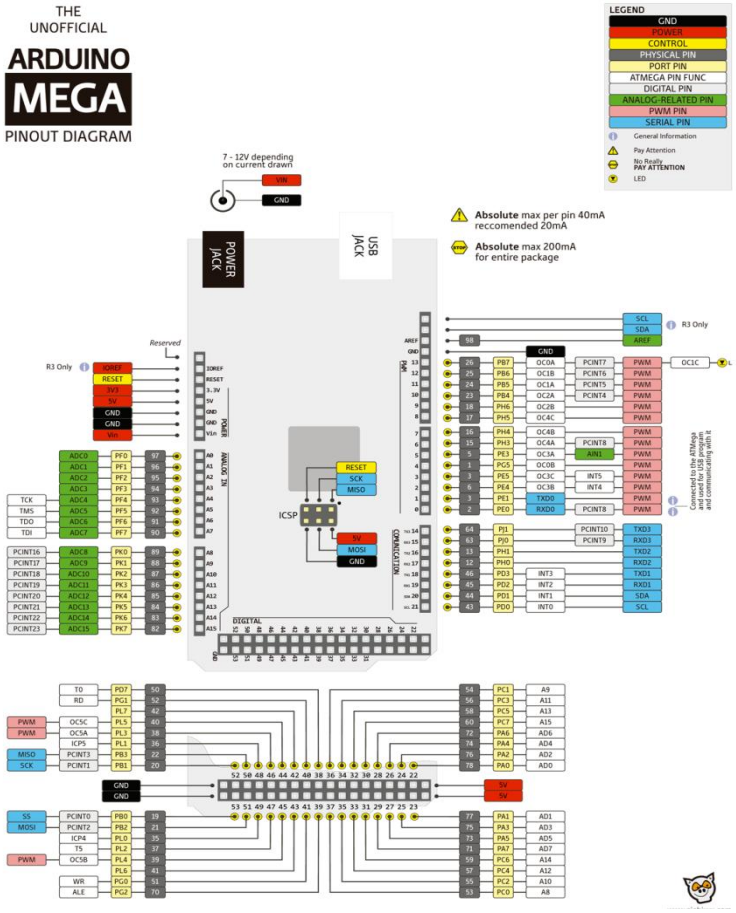


Figura 4.1: Esquema del Arduino Mega 2560.[4]

Aquesta placa està composta per 54 pins entrada/sortida, dels quals 15 són pins PWM (Pulse Width Modulation), 16 sortides analògiques, 4 UARTs (hardware serial ports), un cristall oscil·lador de 16MHz, connexió USB i l'entrada "Jack" de corrent.

L'esquema anterior ve donat per un codi de colors:

Massa
Tensio
Control
Pins Físics
Funció dels pins
Pins digitals
Pins Analògics
Pins PWM
Serial Pins

Taula 4.2: Codi de colors de l'esquema.

4.2.- Mòduls y sensors

En aquest apartat s'explicaran tots els mòduls connectats l'Arduino que donen sentit al software.

El projecte consta de cinc mòduls diferents, que són: El sensor de temperatura i humitat, el sistema Bluetooth, el conjunt de relés, el receptor IR, la pantalla LCD i el rellotge.

Sensor DHT11:

El DHT11 consisteix en un complex sensor de temperatura i humitat amb sortida de senyal digital calibrada. Utilitzant la tècnica exclusiva d'adquisició de senyals digitals i la tecnologia de detecció de temperatura i humitat, garanteix una alta fiabilitat i una excel·lent estabilitat a llarg termini. Aquest sensor inclou un component resistiu per la mesura de la humitat i un component NTC per a la mesura de la temperatura i es connecta a un microcontrolador de 8 bits de gran rendiment, en aquest projecte seria l'ATmega2560, que ofereix una qualitat excel·lent, una resposta ràpida, una capacitat per evitar les interferències i rendibilitat. [Annex II]

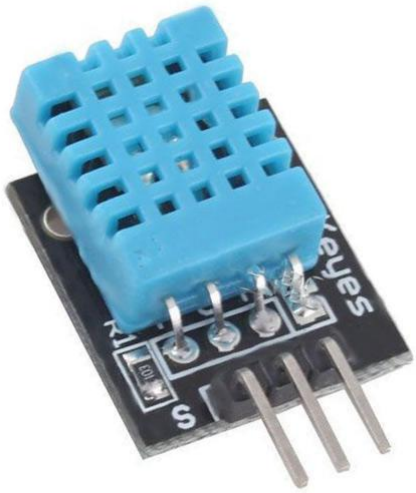


Figura 4.2: Imatge del Sensor DHT11 [5]

El DHT11 té unes característiques especifiques que són:

	Condicions	**Mínim**	**Típic**	**Màxim**
Tensió	DC	3V	5V	5.5V
Corrent	Mesures	0.5mA	---	2.5mA
	Mitjana	0.2mA	---	1mA
	En repòs	100uA	---	150uA

Taula 4.3: Característiques elèctriques del DHT11.

L'esquema típic de connexió del sensor cap al microcontrolador és:

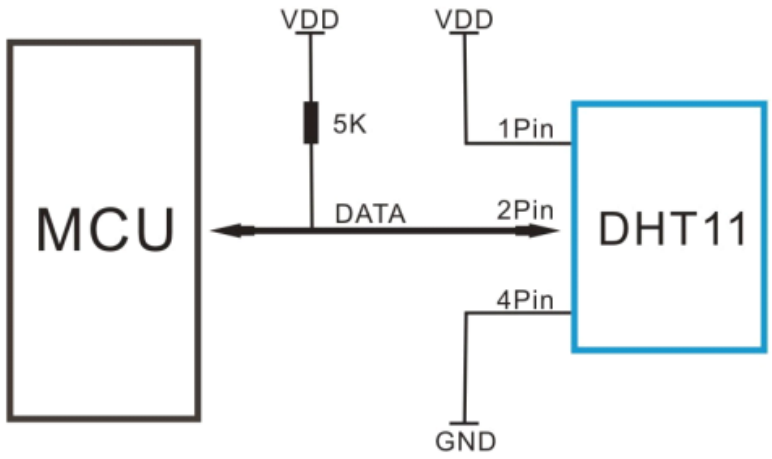


Figura 4.3: Esquema de connexió del sensor DHT11.[5]

El sensor consta de 3 pins de sortida, el pin 1 es connecta a VDD, el pin 2 a una resistència i al microcontrolador i el pin 4 a GND (no és el 3 perquè el sensor consta de 4 pins i el tercer està connectat internament al pin 1, per tant el pin 3 es connecta a VDD). VDD és la tensió d'entrada (5V), GND és massa, MCU és el microcontrolador. Es pot observar una resistència de 5k ohms que actua com a resistència de "pull-up".

Aquest tipus de connexió són utilitzats per evitar lectures errònies quan no el pin del microcontrolador no rebi senyal. En el cas d'aquest projecte, el sensor de temperatura ja porta incorporat aquesta resistència.

Mòdul HC-06

EL HC-06 és un mòdul Bluetooth esclau que utilitza lacomunicació sèrie en el sistema Arduino. Una vegada connectat l'única funció que té és la de transmetre al sistema els valors que rep i enviar els desitjats. [6]

Figura 4.4:Imatge del mòdul HC-06.[7]

El mòdul va integrat a una placa nomenada ZS-040.

Figura 4.5:Imatge del mòdul ZS-040.[7]

Les especificacions tècniques són:

	Condicions	**Mínim**	**Típic**	**Màxim**
Tensió	DC	3.3V	5V	6V
Corrent	Mesures	0.6mA	---	30mA

Taula 4.4: Característiques elèctriques del ZS-040.

Com es pot observar a la figura 5 el mòdul consta de 4 pins:

- Pin 1 RXD: Connexió a TX del Arduino.
- Pin 2 TXD: Connexió a RX del Arduino.
- Pin 3 GND: Connexió a massa.
- Pin 4 VCC: Connexió a VCC.

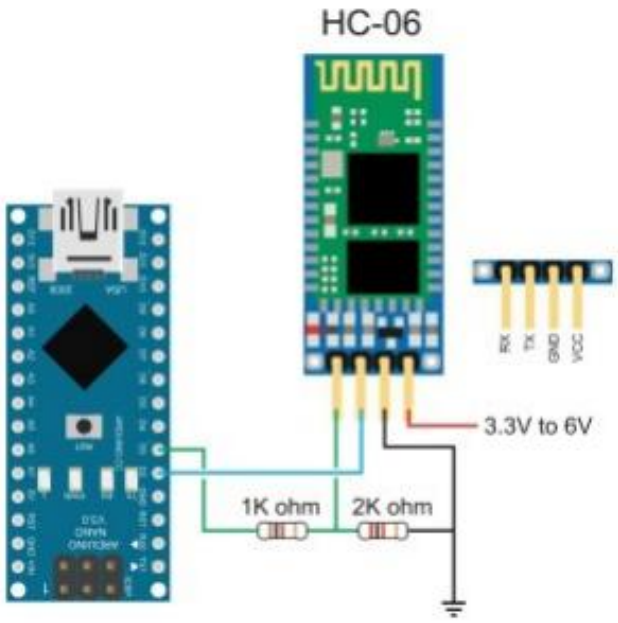


Figura 4.6: Esquema de connexió del mòdul ZS-040.[Annex II]

Relés HL-54S

Un relé és un interruptor operat elèctricament, que permet activar o desactivar un circuit que utilitza tensió i/o corrent molt superiors, al que pot gestionar un microcontrolador. El mòdul HL-54S està format per quatre relés. No hi ha cap connexió entre els circuits de baixa tensió controlada per l'Arduino i el circuit d'alta potència. El mateix relé protegeix circuit l'un de l'altre.

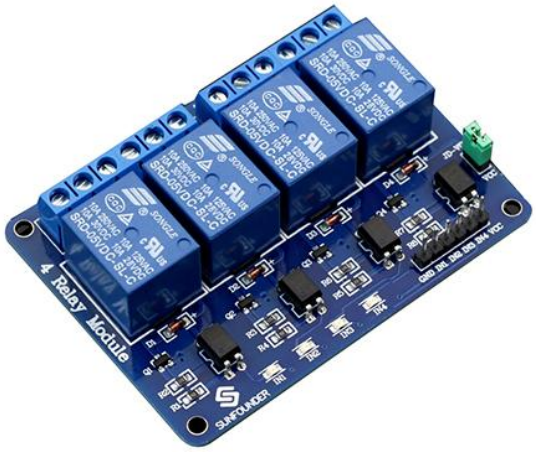


Figura 4.7: Imatge del mòdul HL-45S.[8]

Les especificacions tècniques són:

	Entrada circuit	**Relé (Màxim)**	**Relé (Màxim)**
Tensió	5V DC	250V AC	30V DC
Corrent	0.7 mA	10 A	10 A

Taula 4.5: Característiques elèctriques del HL-45S.

Com es pot observar en la taula anterior aquests relés tenen la capacitat de suportar la tensió en DC i AC i poden ser controlats per un sistema que proporciona 5V.

L'esquema de connexió del HL-45S és:

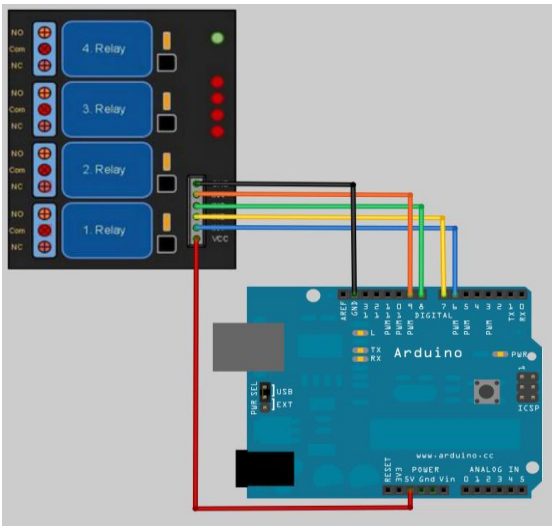


Figura 4.8: Esquema de connexió del mòdul HL-45S.[9]

El conjunt de relés disposa de sis pins de connexió, del 2 al 5 (ambdós inclosos) són els que controlen els quatre relés i van connectats directes a les sortides de l'Arduino. El primer pin és el de l'entrada de tensió de 5V del microcontrolador i l'últim pin és el que va connectat a massa.

Els relés tenen tres connexions cadascun:

- NO: Normalment Obert.
- Com: Comú.
- NC: Normalment Tancat.

La commutació del relé es produeix en el Comú. Depenent l'estat inicial que es vulgui s'haurà de connectar al Normalment obert, quan la necessitat sigui la de mantenir el relé en repòs inicialment, o Normalment Tancat en el cas que es desitgi el contrari.

Receptor IR AX-1838HS

El receptor AX-1838HS és el receptor d'infrarojos del comandament a distància. La seva funció és la de transmetre un valor depenent del botó polsat al comandament.

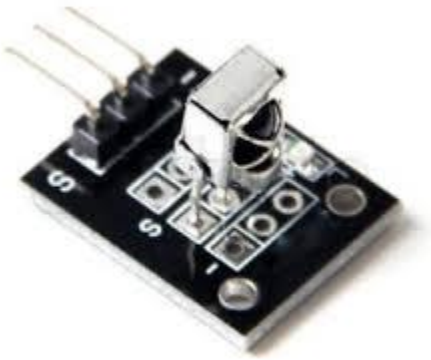


Figura 4.9: Imatge del mòdul d'infrarojos AX-1838HS.[10]

Les especificacions tècniques són:

	Condicions	Mínim	Típic	Màxim
Tensió	DC	2.1 V	5V	5.5V
Corrent	Mesures	---	---	1.5mA

Taula 4.6: Característiques elèctriques del AX-1838HS.

Com es pot observar en la Figura 4.9 el mòdul consta de tres pins, el de la dreta va a massa, el del mig a VCC i el de l'esquerra és el que transporta la informació, per tant, ha de ser connectat al microcontrolador.

Pantalla LCD 1602A

El mòdul 1602A és una pantalla LCD que permet la visualització del text alfanumèric mitjançant la programació en un microcontrolador. Aquesta pantalla té una dimensió de dos línies i una capacitat de setze caràcters per línia.[11]

Figura 4.10: Imatge de la pantalla LCD 1602A.

Les especificacions tècniques són:

	Condicions	Mínim	Típic	Màxim
Tensió	DC	4.7 V	5 V	5.5 V
Corrent	Mesures	---	1.1 mA	---

Taula 4.7: Característiques elèctriques de la pantalla LCD 1602A.

L'esquema de connexió de la pantalla LCD és el següent:

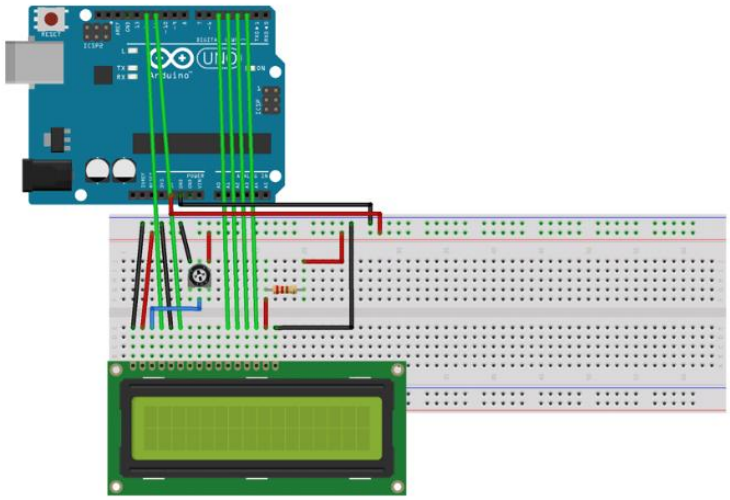


Figura 4.11: Esquema de connexió de la pantalla LCD 1602A.[11]

Com es pot observar en la Figura X.11 no s'utilitzen totes les connexions disponibles en la placa de la LCD.

Els pins usats, ordenats d'esquerra a dreta en la figura anterior, són:

- GND.
- VCC.
- Contrast de la pantalla.
- RS- Selector entre comandos i dades.
- RW- Escriptura i lectura de les comandes i les dades.
- Sincronitzador de la lectura de dades.
- Pins de dades D0-D8 (només s'utilitzen del D5 al D8).
- Alimentació de la llum de la LCD.
- GND de la llum de la LCD.

Per fer la correcta connexió de la pantalla s'ha d'incloure una resistència variable de 10K, per poder calibrar el contrast de la pantalla i una resistència de 330 ohms en l'alimentació de la llum, per regular el voltatge en l'entrada.

Mitjançant un transistor es fa el control de l'alimentació de la llum de la LCD per tal d'apagar-la quan sigui necessari (Mirar en l'annex de connexions).

Rellotge ZS-042

El mòdul ZS-042 és un circuit integral que incorpora el DS3231, que és un rellotge (RTC) de baix cost i extremadament precís amb un oscil·lador de cristall (TXCO). El dispositiu incorpora una entrada de bateria que manté el cronometratge exacte en el cas que s'interrompi l'alimentació principal del dispositiu.

El RTC manté la informació dels segons, minuts, hores, dies, mesos i anys, també incorpora la correcció d'any de traspàs.[12]

Figura 4.12: Imatge del rellotge ZS-042.[12]

Les especificacions tècniques del rellotge són:

	Condicions	**Mínim**	**Típic**	**Màxim**
Tensió	VCC en DC	2.3 V	3.3 V	5.5 V
	Vbat en DC	2.3 V	3 V	5.5V
Corrent	Mesures	---	---	0.2 mA

Taula 4.8: Característiques elèctriques del rellotge ZS-042.

Com es pot observar en la figura 12 el mòdul consta de 6 pins de connexió dels quals només s'utilitzen quatre, que són:

- GND: Connexió a massa
- VCC: Connexió a tensió (5 V)
- SDA: Connexió al pin SDA (sortida de dades) del microcontrolador.
- SCL: Connexió al pin SCL (Rellotge) del microcontrolador.

La connexió VCC és de 5 V, ja que no seria suficient connectar-la a 3 V.

4.3.- Altres elements

En aquest apartat es comentarà els altres elements usats en la part del Hardware que no són mòduls de l'Arduino, però són necessaris a l'hora de fer les connexions.

Aquests elements són: El transistor NPN, les resistències, un ventilador i les llums del terrari.

Transistor PN2222

El PN2222 és un transistor BJT(de l'anglès "bipolar junction transistor") que té la funció d'actuar d'amplificador o commutador, com és el cas d'aquest projecte.[28]

Aquest transistor és del tipus NPN, l'altre tipus que n'hi ha és el PNP, la diferència entre ambdós és el tipus de dopat però la funció és la mateixa.[13]

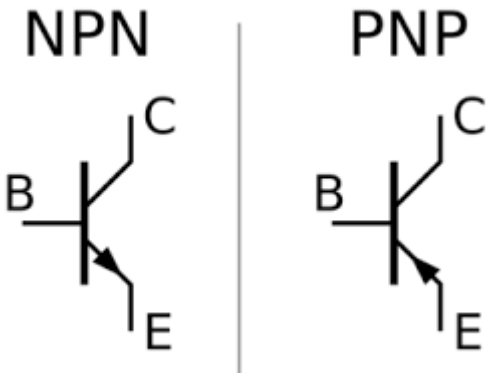


Figura 4.13: Símbols Electronics del dos tipus de transistor BJT .[13]

Les característiques elèctriques del PN2222 són:

	Condicions	**Base**	**Emissor**	**Col·lector**
Tensió	VCC en DC	60 V	30 V	5 V
Corrent		---	---	600 mA

Taula 4.9: Característiques elèctriques del PN2222.

L'esquema del PN2222 és:

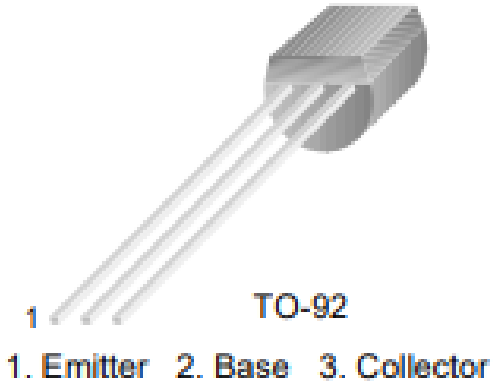


Figura 4.14:Esquema del ordre dels pins que formen el PN2222.[14]

Resistències

La resistència elèctrica és una mesura del grau d'oposició que presenta un objecte al pas del corrent elèctric. La unitat del Sistema Internacional d'Unitats per a la resistència elèctrica és l'ohm, que se simbolitza amb la lletra grega omega majúscula (Ω).[15]

En aquest projecte a part d'utilitzar resistències també es fa ús del potenciòmetre, que és una resistència variable.

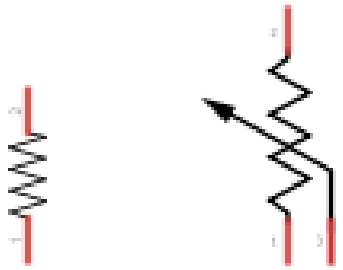


Figura 4.15:Esquema electrònic de la resistència(esquerra)
i del potenciòmetre(dreta).

Ventilador

El ventilador està compost d'un motor que funciona a 5 V en corrent continua. La funció d'aquest ventilador en el projecte és la de refrigerar la caixa on està el microcontrolador, ja que està situada al costat una font de calor (làmpada de calor).

Figura 4.16: Imatge del ventilador.[16]

Il·luminació

El terrari on habita el camaleó està compost per una font de calor i per dos llums, una de radiació UV (ultraviolada) i l'altre és només d'il·luminació en cas que hi hagi molta temperatura fer encendres en lloc de la font de calor.

Aquests elements van dins d'un portalàmpades de porcellana, ja que és més resistent a la temperatura.

Aquests són controlats amb els relés prèviament explicats al apartat 3.2.

Figura 4.17: Imatge del portalàmpades.[17]

4.4.- Costos

Un cop explicats els elements més significatius de la part del Hardware, en aquest apartat s'exposaran tots els costos dels elements que formen el terrari i el sistema incorporat.

Costos inicials

Com a costos inicials s'han de considerar: El camaleó, el terrari, l'ambientació del terrari ("attrezzo"), la font de calor, el llum UV i els dos portalàmpades.

Elements	Quantitat	Cost per unitat (€)
Camaleó	1	40,00
Terrari	1	80,90
"Attrezzo" *	1	52,72
Font de calor	1	25.63
Llum UV	1	9,26
Portalàmpades	2	22,50
Total		**231.01**

* El attrezzo no són tots iguals i son de diversos preus per tant es contarà la suma total.

Taula 4.10: Costos inicials.

Els costos inicial és de 231,01 € aproximadament, sense tenir en compte el preu de l'alimentació del animal, ja que és un cost variable i irrellevant pel projecte.

Costos finals

Un cop calculats els costos inicials es donarà pas a calcular els costos del projecte, que inclou tota la part del hardware explicat anteriorment, el qual implica tots els elements de l'Arduino i l'ampliació de la llum extra.

Elements	Quantitat	Cost per unitat (€)
Arduino Mega	1	13,99
DHT11	1	2,42
HC-06	1	9,98
HL-45	1	4,80
AX-1838 HS	1	6,41
LCD1602-A	1	5,95
ZS-042	1	1,54
PN2222	2	0,01
Ventilador	1	4,68
Resistències*	5	0,00
Portalàmpades	1	22,50
Làmpada	1	5,46
Cables	64	0,06
Caixa	1	25,87
Placa de talps	1	6,87
Connexions PCB	4	0,35
Total del projecte		**115,73**
Costos inicials		231,01
Total		**346,74**

* El valor de les resistències es contarà com a nul, ja que el valor de 5 resistències està per sota d'1 cèntim.

Taula 4.11: Costos finals.

El cost total és de 346,74 euros aproximadament, ja que no s'ha tingut en compte el valor de les resistències ni l'estany usat per soldar, i més elements dels quals ja es disposaven.

5.- Pressupost i Impacte Ambiental

Un cop vistos tots els costos és proposa un pressupost i l'impacte ambiental del sistema creat.

5.1.-Pressupost

El pressupost d'aquest projecte està format pel conjunt dels costos que formen els elements del hardware, explicats en l'apartat anterior, que donen forma al sistema i el conjunt d'hores treballades per a arribar al resultat final.

Primer es calcula el cost de les hores treballades:

Hores treballades	Preu per hora	Total(€)
546	20	10.920

Taula 5.1: Taula del preu d'hores treballades.

El nombre d'hores treballades està en l'apartat de Planificació i s'obté fent una mitjana entre el màxim i el mínim d'hores establertes.

Elements	Quantitat	Cost per unitat (€)
Arduino Mega	1	13,99
DHT11	1	2,42
HC-06	1	9,98
HL-45	1	4,80
AX-1838 HS	1	6,41
LCD1602-A	1	5,95
ZS-042	1	1,54
PN2222	2	0,01
Ventilador	1	4,68
Cables	64	0,06
Caixa	1	25,87
Placa de talps	1	6,87
Connexions PCB	4	0,35
Total elements		**87.77**
Preu Hores Treballades		**10.920**
Pressupost Total		**11.007,77**

Taula 5.2: Taula del pressupost.

Aquest valor d'11.007,77 és el preu que té el projecte. Per poder posar-ho a la venda s'haurien de reduir costos, ja que el cost del sistema és de 87,77 €, un preu molt elevat. Es preveu que la reducció del cost final pot ser fins al 50%, ja que els mòduls de l'Arduino estan encarits al ser comprats en plaques ja soldades i si es fes una única placa del sistema, es podria estalviar la caixa i el ventilador. S'estima que es podria arribar a un cost de fabricació al voltant dels 45 €, incloent-hi el cost de fabricació per unitat. Al preu final se li suma un 30% del marge comercial, és a dir, els beneficis.

Càlcul del cost per unitat	
Concepte	**Cost(€)**
Cost del prototip	45
Marge comercial 30%	13.5
Preu venda unitari	**58.5**

Taula 5.3: Taula del càlcul del cost per unitat.

Utilitzant el marge comercial, és a dir, els beneficis per unitat, s'arriba a la conclusió que a partir de la unitat 816 venuda es començarà a obtenir guanys.

Beneficis	
Concepte	**Cost(€)**
Beneficis	13.5
Unitats venudes	816
Beneficis Totals	**11.016**

Taula 5.4: Taula dels beneficis.

Es pot observar en l'anterior taula que Beneficis Totals > Pressupost Total.

A partir d'aquest moment, el marge comercial es pot reduir a un 20%.

Càlcul del cost per unitat final	
Concepte	**Cost(€)**
Cost del prototip final	45
Marge comercial 20%	9
Preu venda unitari final	**54**

Taula 5.5: Taula del càlcul del cost per unitat final.

A partir de les 816 unitats venudes, el preu de venda final es reduiria a 54 € per unitat.

5.2.-Estudi de mercat

En el cas de comercialitzar aquest producte com a empresa, s'ha de realitzar un estudi de mercat. Es tracta de detallar el mercat en el qual l'empresa mantindria l'activitat principal, així com els clients potencials i la competència.

El sector on es desenvoluparia aquesta empresa seria en el sector d'animals exòtics terrestres. Segons dades recollides per ANFAAC, en el 2017, a Espanya hi havia més d'un millor de rèptils com a animals de companyia. Per tant aquest seria el mercat del producte. Els possibles clients podrien ser tant en tendes d'animals o particulars, ja que cada cop hi ha més gent que adquireix aquest tipus d'animals. [27]

En les tendes el producte podria ser útil per tenir motoritzat tot el control dels valors de temperatura i humitat, i dels horaris de llum de tots els animals en venda.

Per qualsevol particular aquest sistema suposaria una ajuda a l'hora del manteniment de l'animal .

Aquest sistema també podria ser útil per aquelles persones que tinguin un treball nocturn, ja que no haurien s'haurien de preocupar per la correcta administració d'hores de llum.

S'ha fet una recerca a diverses tendes en línia i físiques, i actualment no hi ha en venda productes semblants o iguals, per tant podria ser un mercat al qual adherir-se.

Una altra sortida de mercat podria ser com a kit educatiu per a escoles d'enginyeria. En els graus podria ser útil per aquelles assignatures que envoltin la programació de microcontroladors. Es podria utilitzar el kit com a pràctiques de l'assignatura, a cada pràctica s'aniria afegint un mòdul diferent i finalment, el professor amb l'aplicació podria comprovar el funcionament.
En les assignatures de creacions d'aplicacions mòbils es podria fer el procés contrari, en tenir el kit es podria crear una aplicació per fer el control.

5.3.-Impacte Ambiental

Un estudi d'impacte ambiental és el document resultant de l'anàlisi de les possibles conseqüències d'un projecte sobre la salut del medi ambient i dels ecosistemes.
Contingut de l'estudi:

- *Descripció del projecte*: El sistema creat consisteix en el control d'un terrari. Incorpora diversos elements que provoquen petits impactes ambientals. Els elements del sistema que poden ocasionar impactes són: el microcontrolador, el ventilador, el sensor de temperatura i els relés.

- *Descripció del medi*: El projecte és instal·lat dins d'un terrari artificial, dins d'una casa. L'espai afectat en aquest cas és la part superior del terrari i l'interior d'ell que és on s'incorpora el sensor de temperatura.

- *Identificació i caracterització d'impactes:* El microcontrolador i l'electrònica que l'envolta provoquen una pèrdua de calor per efecte Joule, és mínima. El ventilador i els relés provoquen contaminació acústica, més el ventilador que els relés, ja que aquests últims només fan soroll en fer el canvi d'estat, per altra banda el ventilador fa un soroll constant a l'estar connectat. El sensor de temperatura provoca una contaminació visual, ja que està dins del terrari del camaleó el qual hi ha vegetació dins, la contaminació és mínima. Existeix una contaminació lumínica dels leds incorporats a diversos mòduls. La contaminació lumínica de dia és mínima però de nit és més apreciable.

- *Mesures correctores:* El ventilador és la mesura correctora de la pèrdua de calor. Per eliminar la contaminació lumínica es poden tapar els leds, i per solucionar o reduir la contaminació acústica es pot limitar l'obertura del ventilador a quan no hi hagi ningú a prop.

6.- Software

Els programes utilitzats per a la programació del sistema són: Android Studio i Arduino IDE per a l'aplicació i la placa de desenvolupament, respectivament.

La finalitat d'aquest apartat és per explicar resumidament el funcionament dels programes usats i algunes de les seves funcions per així fer més fàcil l'explicació del codi, en els apartats 7 i 8.

Android Studio

L'Android Studio és l'entorn de desenvolupament oficial per a la plataforma Android, i està disponible gratuïtament per a les plataformes Microsoft Windows, macOS i Linux.

Figura 6.1.- Logotip de l'Android Studio.[18]

El llenguatge utilitzat és el Java per a la programació de les funcions i l'XML per a la programació visual.

L'aplicació creada està pensada per dispositius amb la versió Android 7.0, ja que és l'actualització del telèfon utilitzat. Això implica que aquesta app no hauria de funcionar correctament en un dispositiu amb un sistema operatiu Android menor a aquest.[19]

Una aplicació creada per Android Studio conté una sèrie arxius diferents, i a continuació s'expliquen els més rellevants:

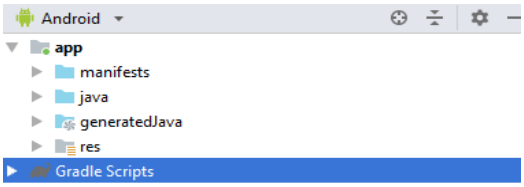


Figura 6.2.- Carpetes de l'aplicació.

Com es pot observar en la figura anterior, en crear una aplicació, s'introdueixen una sèrie de carpetes per defecte. Les més rellevants i les que s'han modificat són: Manifests, Java i Res.

Manifests:

En obrir aquesta carpeta s'observa l'arxiu AndroidManifest.xml.

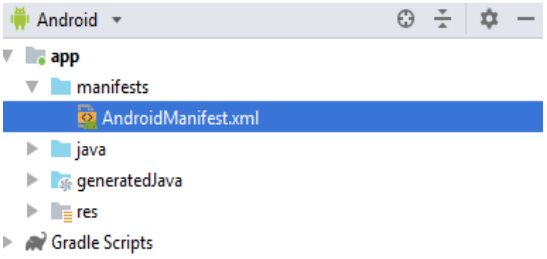


Figura 6.3.- Carpeta manifests.

Aquest arxiu conté la configuració bàsica de l'app. Incorpora el nom de l'aplicació, la imatge que es mostrarà al menú del mòbil, els permisos necessaris de funcions del telèfon i l'ordre d'execució dels arxius Java, entre altres coses.

Java:

Aquesta carpeta és la localització dels arxius on es duen a terme les funcions de l'aplicació.

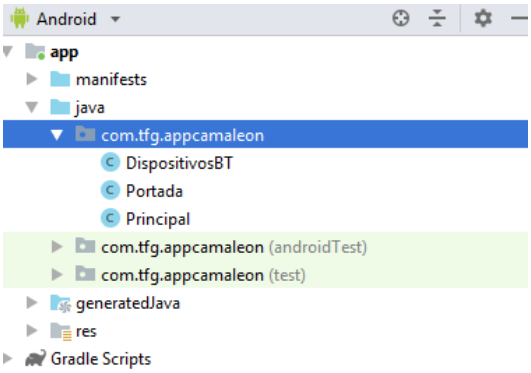


Figura 6.4.- Carpeta java.

Aquí incorpora tres arxius diferents, DispositivosBT, Portada i Principal. Aquestes activitats contenen totes les funcions de l'aplicació. Portada és la primera a ser executada, tot seguit del DispositivosBT, on s'inicia la comunicació i Principal que és on està el panell de control sobre el microcontrolador. El codi d'aquests arxius són explicats amb més detall en l'apartat 8.

Res:

El contingut d'aquesta carpeta és tot el relacionat amb tema visual de l'aplicació, és on hi ha més arxius.

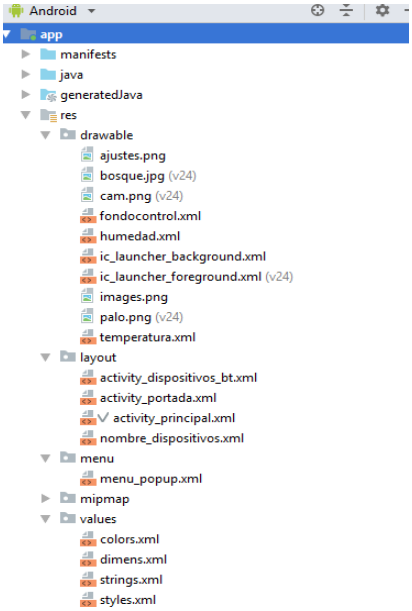


Figura 6.5.- Carpeta res.

La primera carpeta, drawable, conté totes les imatges i els arxius que s'executen com a imatge.

El layout és on resideixen els arxius de programació visual de les pantalles, i aquests estan directament relacionats amb els arxius de la carpeta java.

La carpeta menú conte el popup que apareix en la portada en polsar el camaleó.

L'última conté els colors ordenats perquè sigui més fàcil la seva utilització, les dimensions de la pantalla, tots noms utilitzats per a cada objecte utilitzat en els arxius layout i finalment l'últim arxiu correspon als estils de la pantalla.

En els arxius de Java, l'Android Studio executa diverses funcions segons el moment d'execució de l'aplicació, això s'anomena el cicle de vida i és:

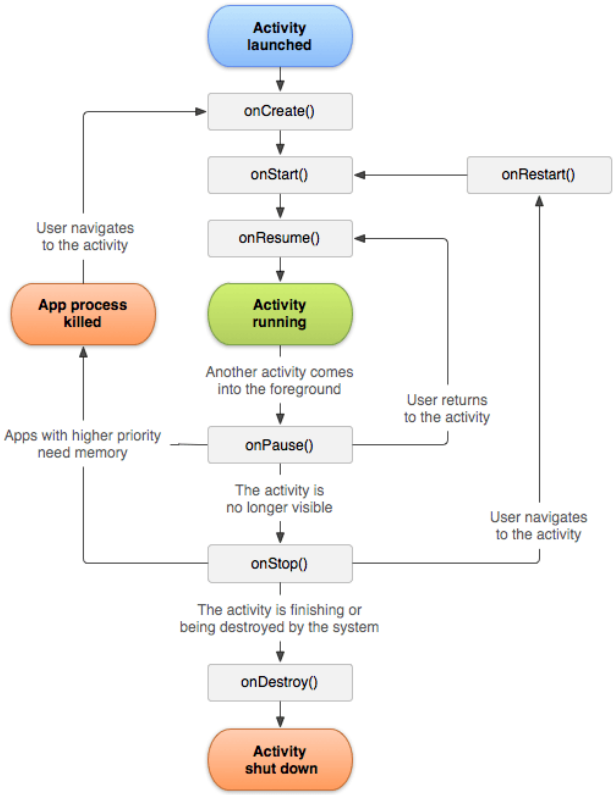


Figura 6.6.- Cicle de vida d'una activitat a Android Studio.[20]

onCreate: Primer mètode que es crida en iniciar l'activitat. És l'activitat que s'executa abans que l'usuari pugui veure la pantalla.

onStart: Aquest mètode és executat just abans que l'activitat sigui visible per l'usuari. S'utilitza per recarregar valors o de pont per a "frame rates"(Per a la construcció de jocs).

onPause: És cridat just abans de canviar a una altra activitat. S'utilitza per a confirmar canvis no guardats, per netejar o destruir recursos que consumeixen i altres funcions. No es pot incorporar funcions de llarga duració, ja que no finalitzarà l'activitat fins que no acabi la funció.

onResume: S'executa quan l'activitat està a punt de començar a interactuar amb l'usuari després d'estar en un estat de pausa.

onStop: Aquest mètode s'executa en el moment que l'activitat no és visible per l'usuari. Aquest esdeveniment pot succeir perquè l'activitat hagi finalitzat, perquè el sistema estigui destruint l'activitat per estalviar recursos o per un canvi d'orientació del dispositiu.

onRestart: Un cop l'activitat s'hagi parat, abans que s'iniciï de nou, s'executa aquesta activitat. Aquest mètode sempre és seguit per l'onStart. Només passarà en el cas que sigui necessari realitzar alguna funció prèvia a l'onStart.

onDestroy: Aquest és l'últim mètode cridat abans que l'activitat sigui destruïda, és a dir, en el moment que és canvi d'activitat. Després de ser cridada aquesta, l'activitat finalitzarà i els recursos seran purgats.

onSaveInstanceState: Aquest mètode proporcionat per Android dona l'oportunitat de guardar dades quan es produeix un canvi, per exemple, quan es produeix un canvi en l'orientació de la pantalla.[20]

Arduino IDE

L'entorn de desenvolupament integral d'Arduino és una aplicació multiplataforma, per a Windows, Mac i Linux, que s'utilitza per escriure i carregar programes a taulers compatibles amb Arduino. El llenguatge de programació és el Java.

Figura 6.7.- Logotip de l'Arduino IDE.[21]

La placa utilitzada per aquest projecte és un Arduino Mega 2560, junt amb els variïs mòduls connectats que donen forma a la part física del projecte. Mitjançant la programació en l'Arduino IDE s'arriba al control d'aquests mòduls.

L'estructura de programació d'aquest entorn de desenvolupament és la següent:

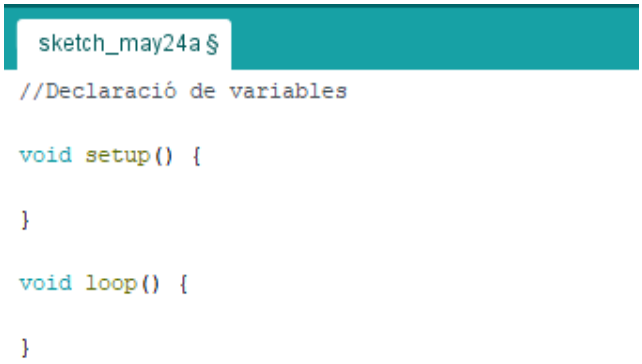


Figura 6.8.- Exemple d'un programa nou d'Arduino.

En iniciar un nou fitxer apareix com en la figura 6.8.

La part inicial s'utilitza per crida les llibreries que són utilitzades pels mòduls.

Una peculiaritat que té aquest sistema de programació, és la incorporació de llibreries específiques per a cada mòdul connectat, que faciliten la programació. Normalment aquestes llibreries les donen els fabricants.

Un cop declarades les variables i les llibreries es dóna pas a la funció setup, la qual només s'executa una vegada. Aquesta funció és ideal per a iniciar l'estat (ON/OFF) de les entrades i sortides de la placa, iniciar la comunicació i les diverses característiques de cada mòdul connectat, si és el cas.

L'última funció, loop, és aquella que està en continua execució, és a dir, quan acaba totes les funcions, torna a començar-les. En aquesta part és on es programa totes les funcionalitats, ja sigui lectura d'algun sensor com el canvi d'estat de les sortides de la placa, com per exemple el control d'uns relés.

6.1.-Diagrama de blocs del sistema

A continuació és mostrara el diagrama de blocs del sistema per així facilitar l'entendiment del codi del programa. En aquest diagrama apareixen els següents elements.

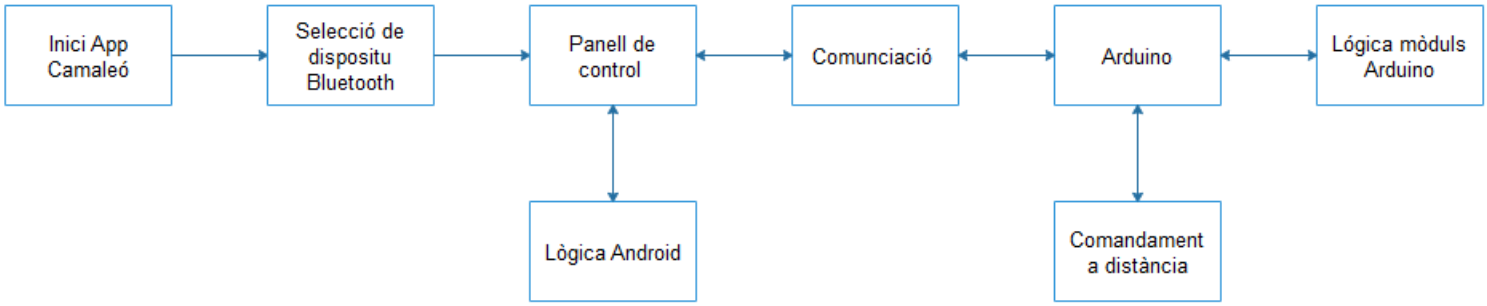


Figura 6.9.- Diagrama de blocs del sistema.

La part de l'esquerra fins al bloc de comunicació correspon a l'aplicació mòbil i la part dreta correspon a l'Arduino. Quan es parla de lògica es fa referència als conjunts de funcions i accions de les aplicacions.

7.- Explicació codi Android

Un cop explicat resumidament la funcionalitat de l'Android Studio, és moment de l'explicació del codi usat, com ja s'ha explicat, aquesta app està formada per fitxers Java i XML, els considerats més rellevants pel funcionament final. En l'Annex I estan inclosos tots els fitxers. Aquesta aplicació és funcional per dispositius Android amb un sistema operatiu superior al 7.0. El dispositiu també ha d'incorporar comunicació Bluetooth, si no en té, no podrà connectar-se a l'Arduino i per tant no funcionarà.

A continuació els diagrames de flux són exposats i s'explica el funcionament de les parts més importants dels fitxers Java, que són: Portada, DispositivosBT i Principal.

Per facilitar l'enteniment d'aquestes funcion, en la figura 7.1 és mostra un diagrama de blocs del funcionament dels tres fitxers.

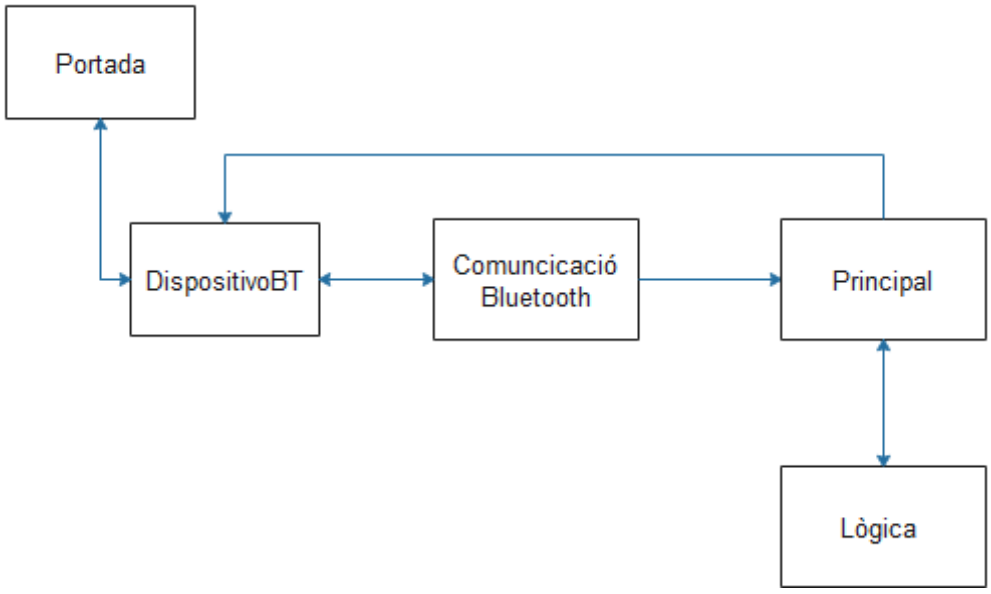


Figura 7.1.- Diagrama de blocs Android.

7.1.-Portada

Digrama de flux

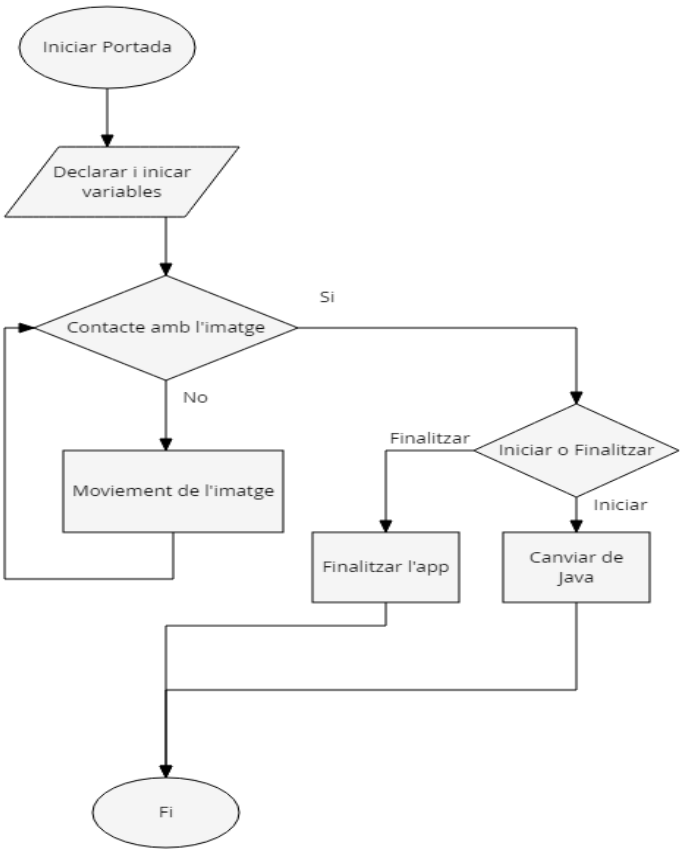


Figura 7.2.- Diagrama de flux de Portada.

Codi

A continuació s'explicarà detalladament el Java Portada, que seguint el diagrama de blocs de l'apartat anterior, dóna lloc a la primera pantalla de l'aplicació.[22]

Primerament es troba la funció package, que mostra la carpeta on es troba el document Java en qüestió dins de l'aplicació.

```
package com.tfg.appcamaleon;
```

A continuació s'importen les llibreries que s'utilitzaran posteriorment per al correcte funcionament de les funcions en ús.

```
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
```

```
import android.os.Bundle;
import android.view.MenuItem;
import android.view.View;
import android.widget.ImageView;
import android.os.CountDownTimer;
import android.widget.PopupMenu;
```

Tot seguit es crea la classe Portada, que és el lloc on van totes les funcions d'aquest fitxer Java i on es declaren les variables a utilitzar.

```
public class Portada extends AppCompatActivity {
ImageView Camaleon;
double cont = 0;
double Cont2 = 0, angulo = 0;
int Valor=0;
```

Un cop inicialitzada les variables es defineix la funció onCreate..

```
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_portada);
```

A continuació s'enllacen les variables declarades amb les del fitxer XML corresponent. En aquest cas el fitxer és el activity_portada.xml(Annex I).

```
Camaleon = (ImageView) findViewById(R.id.idFotoCamaleon);
```

Tot seguit són creats una sèrie de cronòmetres, en total quatre, per tal de fer la simulació del moviment d'una imatge d'un camaleó. Per fer-ho s'utilitzen un conjunt de CountDownTimers.

En aquesta funció és necessària la introducció de dos temps en mil·lisegons, el primer és el temps final del cronòmetre (color verd) per la funció onFinish i el segon és el temps d'interval de compte enrere (Color vermell) per la funció onTick.

```
new CountDownTimer(2600, 100){

 void onTick(long millisUntilFinished) {}

public void onFinish() {}

}
```

Dins de la següent funció és on es crea el moviment calculat prèviament.

```
public void onTick(long millisUntilFinished) {
cont--;
Cont2 = Cont2 - 0.61538;
Camaleon.setTranslationX((float) cont);
Camaleon.setTranslationY((float) Cont2);
}
```

En la funció onFinish és on es tornen a inicialitzar les variables i on es crea el següent CountDownTimer per tal que segueixi el moviment.

```
public void onFinish() {
    cont = Camaleon.getTranslationX();
    Cont2 = Camaleon.getTranslationY();
    angulo = Camaleon.getRotation();
    new CountDownTimer(8200, 100) {
       public void onTick(long millisUntilFinished) {
       cont--;
       angulo = angulo - 0.244;
       Cont2 = Cont2 - 0.36;
        }
```

L'acció setTranslation té la funció de desplaçar el numero introduït.

```
Camaleon.setTranslationX((float) cont);
Camaleon.setTranslationY((float) Cont2);
```

L'acció setRotation té la funció de rotar la imatge el numero introduït.

```
Camaleon.setRotation((float) angulo);
}
public void onFinish() {
cont = Camaleon.getTranslationX();
Cont2 = Camaleon.getTranslationY();
angulo = Camaleon.getRotation();
```

Un cop finalitzat el moviment es deixa l'últim onFinish buit per a evitar un error de programació i tot seguit es col·loca l'acció .start() al final de cada cronòmetre, per al seu correcte funcionament.

```
public void onFinish() {

}
}.start();
```

La següent línea de codi s'executa en el moment que es polsa l'imatge del camaleó.[23]

```
Camaleon.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) {
}
```

Un cop dins de la funció onClickListener és cridat el PopUpMenu, que prové d'un fitxer XML anomenat menú_popup.xml (Consular en Annex I). El qual un cop polsat la imatge del camaleó apareixerà un menú.[24]

```
final PopupMenu popupMenu = new PopupMenu(Portada.this,
Camaleon);
popupMenu.getMenuInflater().inflate(R.menu.menu_popup,
popupMenu.getMenu());
```

Un cop aparegui el menú, sortiran dues opcions a escollir, iniciar o sortir. L'opció escollida es guardarà en la variable item.

```
popupMenu.setOnMenuItemClickListener(new
PopupMenu.OnMenuItemClickListener() {
@Override
public boolean onMenuItemClick(MenuItem item) {}
```

Depenent de la selecció, es farà una acció o un altre, mitjançant la funció switch.

```
switch (item.getItemId()){}
```

En cas d'haver escollit l'opció iniciar, l'acció serà la de canviar d'activiat Java.

```
case R.id.Configuracion:
     Intent Cambio=new
     Intent(Portada.this,DispositivosBT.class);
     startActivity(Cambio);
     break;
```

En cas d'haver escollit l'opció sortir es finalitzarà l'aplicació.

```
case R.id.Salir:
     finish();
     break;
```

En cas de no escollir cap opció o tocar qualsevol altra part de la pantalla on no sigui el menú, es tancarà.

```
return false;
}
});
popupMenu.show();
```

7.2.- DispositivosBT

Digrama de flux

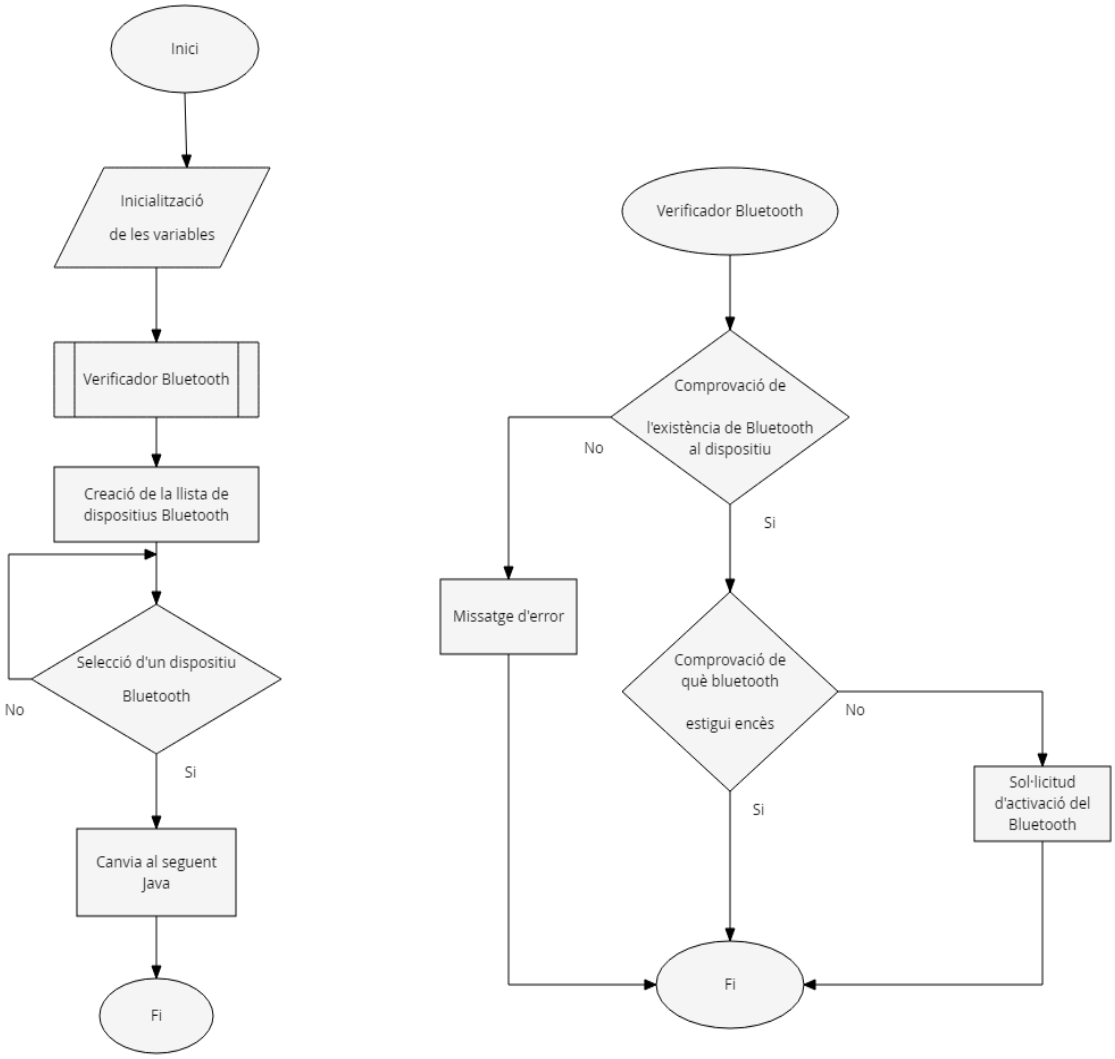


Figura 7.3.- Diagrama de flux de DispositivosBT.

Aquest diagrama de blocs està compost per dos diagrames, el de l'esquerra és el diagrama de la funció principal i el de la dreta forma part d'una subrutina per comprovar l'estat del Bluetooth, la qual és cridada al diagrama principal.

Codi

A continuació s'explicarà el codi usat en l'arxiu DispositivosBT.java. De la mateixa manera que en el codi Portada, es comença amb la localització dels arxius dins del projecte i la importació de les llibreries que són utilitzades.[6]

```
package com.tfg.appcamaleon;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.content.Intent;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import java.util.Set;
```

Tot seguit es declara la classe DispositivioBT i les variables que són utilitzades, i s'enllaça amb l'arxiu activity_dispositivos_bt.xml, que és creat simultàniament amb l'arxiu Java, en la funció onCreate.

```
public class DispositivosBT extends AppCompatActivity {

    private static final String TAG = "DispositivosBT";
    ListView IdLista;
    public static String EXTRA_DEVICE_ADDRESS =
    "device_address";
    private BluetoothAdapter mBtAdapter;
    private ArrayAdapter mPairedDevicesArrayAdapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_dispositivos_bt);
    }
```

Un cop enllaçat, es crea la funció onResume que s'executa just després que l'onCreate. Tot seguit es crida a la subrutina per verificar l'estat del Bluetooth (està explicat més endavant en aquest mateix apartat).

```
@Override
public void onResume()
{
    super.onResume();
     VerificarEstadoBT();

}
```

Es declaren i s'enllacen les variables a utilitzar. La primera variable s'enllaça a l'arxiu nombre_dispositivos.xml per a fer una llista amb els dispositius Bluetooth.
També s'obté l'adaptador Bluetooth local i s'obtenen els dispositius ja aparellats.

```
mPairedDevicesArrayAdapter = new ArrayAdapter(this,
R.layout.nombre_dispositivos);
IdLista = (ListView) findViewById(R.id.idLista);
IdLista.setAdapter(mPairedDevicesArrayAdapter);
IdLista.setOnItemClickListener(mDeviceClickListener);
mBtAdapter = BluetoothAdapter.getDefaultAdapter();
Set <BluetoothDevice> pairedDevices =
mBtAdapter.getBondedDevices();
```

Tot seguit es comprova que el valor de dispositius aparellats sigui major a 0 (que hi hagi algun) i s'omple la llista amb el nom del dispositiu i la direcció MAC d'aquest.[25]

```
if (pairedDevices.size() > 0)
{
    for (BluetoothDevice device : pairedDevices) {

        mPairedDevicesArrayAdapter.add(device.getName() +
"\n" + device.getAddress());
    }
}
```

Un cop estigui la llista amb els dispositius aparellats, en seleccionar el dispositiu Bluetooth HC-06, s'executarà la funció per fet anar al panell de control (Portada.java).
Però primer extreu l'adreça MAC que són els últims 17 caràcters de la View i després amb la funció intent passa a la següent activitat.

```
private AdapterView.OnItemClickListener mDeviceClickListener =
new AdapterView.OnItemClickListener() {

public void onItemClick(AdapterView av, View v, int arg2, long
arg3) {

        String info = ((TextView) v).getText().toString();
        String address = info.substring(info.length() - 17);

        Intent i = new Intent(DispositivosBT.this,
Principal.class);

        i.putExtra(EXTRA_DEVICE_ADDRESS, address);
        startActivity(i);

    }
};
```

Per acabar amb aquesta activitat, es comenta la subrutina per verificar el Bluetooth del dispositiu. Primer comprova que el dispositiu contingui Bluetooth mitjançant la condició if i després comprova que estigui encès, en el cas que alguna condició no es compleixi, sortirà un missatge d'alerta.

```
private void VerificarEstadoBT() {
    // Comprueba que el dispositivo tiene Bluetooth y que está
encendido.
mBtAdapter= BluetoothAdapter.getDefaultAdapter();
if(mBtAdapter==null) {
  Toast.makeText(getBaseContext(), "El dispositivo no  soporta
Bluetooth",    Toast.LENGTH_SHORT).show();
} else {
        if (mBtAdapter.isEnabled()) {
            Log.d(TAG, "...Bluetooth Activado...");
        } else {
            //Solicita al usuario que active Bluetooth
          Intent enableBtIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
            startActivityForResult(enableBtIntent, 1);

        }
    }
}
```

7.3.-Principal

En aquest apartat s'explicarà l'últim arxiu de l'Android Studio, el qual és el més extens i en el que hi ha la part més important, on s'estableix la connexió i es duen a terme totes les accions amb el microcontrolador.

En ser el codi més llarg, serà dividit en diversos apartats, perquè sigui més aclaridor.

Diagrama de flux

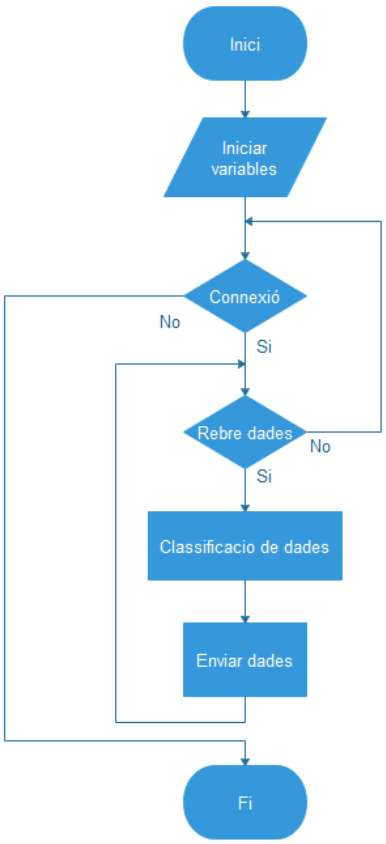


Figura 7.4.- Diagrama de flux de Principal.

Codi

Abans de començar amb les a explicar les funcions, es comença amb la localització de l'arxiu dins de la carpeta de l'aplicació, s'importen les llibreries que s'utilitzaran, es creen i s'inicialitzen les variables.

```
package com.tfg.appcamaleon;

import android.content.Intent;
import android.os.CountDownTimer;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.CheckBox;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.os.Handler;
import android.widget.Toast;
import android.widget.ToggleButton;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.UUID;
import static java.lang.Boolean.FALSE;
import static java.lang.Boolean.TRUE;
public class Principal extends AppCompatActivity {
Text
ViewSens,Otros,Temp,Hum,Hora,Minutos,CalorApagado,PrincipalApagado,Sec
undarioApagado  ,Encendido;

    ProgressBar Prim,Terc,TemperaturaBar, HumedadBar;
    ImageView Ventilador;
    CheckBox Comp;
    ImageView Conf;
    ImageButton SetHora;
    ToggleButton Luz,Luz2,Calor;
    String a,b,c;
    double MovimientoTemp,MovimientoHum;
    int Horacheck,Mincheck,Calordorm,Encender;
    float Cont;
    //Comunicacion
    public Handler bluetoothIn;
    public final int handlerState = 0;
    private BluetoothAdapter btAdapter = null;
    private BluetoothSocket btSocket = null;
    private StringBuilder DataStringIN = new StringBuilder();
    private ConnectedThread MyConexionBT;
    private static final UUID BTMODULEUUID =
UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");
    private static String address = null;
```

OnCreate

Un cop importades les llibreries, la primera funció és l'onCreate, el qual és la primera funció que s'executa en iniciar l'arxiu.

Primer es crea la funció onCreate, s'enllaça amb l'arxiu activity_principal.xml i després es relacionen les variables creades anteriorment amb les corresponents a l'XML corresponent.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_principal);
    Ventilador=(ImageView) findViewById(R.id.idVentilador);
    Sens=(TextView)findViewById(R.id.Sensores);
    Temp=(TextView)findViewById(R.id.Temperatura);
    Hum=(TextView)findViewById(R.id.idHumedad);
    Hora=(TextView)findViewById(R.id.idHora);
    Minutos=(TextView)findViewById(R.id.idMinutos);
    CalorApagado=(TextView)findViewById(R.id.idCalorApagado);
    PrincipalApagado=(TextView)findViewById(R.id.idPrincipalApagado);

SecundarioApagado=(TextView)findViewById(R.id.idSecundariaApagado);
    Encendido=(TextView)findViewById(R.id.idEncender);
    Prim=(ProgressBar)findViewById(R.id.ProgressBar);
    TemperaturaBar=(ProgressBar)findViewById(R.id.idTemperaturaBar);
    HumedadBar=(ProgressBar)findViewById(R.id.idHumedadBar);
    Comp=(CheckBox)findViewById(R.id.idComp);
    Conf=(ImageView)findViewById(R.id.idAjustes);
    Calor=(ToggleButton) findViewById(R.id.idCalor);
    Luz=(ToggleButton)findViewById(R.id.idLuz);
    Luz2=(ToggleButton)findViewById(R.id.idLuz2);
    SetHora=(ImageButton)findViewById(R.id.idHoraEnviar);
```

Tot seguit apareix el primer del botó, Conf, el qual la seva funció és la de tornar al DispositivosBT.

```
Conf.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent Change = new
Intent(Principal.this,DispositivosBT.class);
        startActivity(Change);
        finish();
    }
});
```

Els següents tres botons que pareixen són els de les llums. L'única funció que tenen és la d'enviar una dada mitjançant la comunicació Bluetooth, després aquesta dada serà processada al microcontrolador.

```
Calor.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {MyConexionBT.write("Q");
    }
});
Luz.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {MyConexionBT.write("W");
    }
});
Luz2.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) { MyConexionBT.write("R");
    }
});
```

Les següents línies de codi corresponen al botó de configuració de l'hora d'apagada dels llums del terrari. Primerament es comprova que el valor de l'hora sigui el correcte o no estigui buit.[26]

```
SetHora.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

   if (Hora.getText().length()<2 || Minutos.getText().length() < 2) {
         Toast.makeText(getBaseContext(), "El formato no es correcto",
Toast.LENGTH_LONG).show();
            Hora.setText("");
            Minutos.setText("");
            a = "";
            b = "";
        }
```

Si aquesta condició es compleix, es procedeix a comprovar que els valors de les hores són correctes, dins del rang de 00:00 fins a 24:00, si no és així, s'informa mitjançant un missatge, i en el cas contrari s'envia el valor de les hores, dada a dada, ja que el mòdul HC-06 només pot rebre caràcters individuals.

```
else {
    a = String.valueOf(Hora.getText().subSequence(0, 1)) +
String.valueOf(Hora.getText().subSequence(1, 2));
    b = String.valueOf(Minutos.getText().subSequence(0, 1)) +
String.valueOf(Minutos.getText().subSequence(1, 2));
    c = a + b;
    Horacheck = Integer.parseInt(a);
    Mincheck = Integer.parseInt(b);

    if (Horacheck < 24 && Mincheck < 60) {
```

```
MyConexionBT.write(String.valueOf(Hora.getText().subSequence(0, 1)));

MyConexionBT.write(String.valueOf(Hora.getText().subSequence(1, 2)));

MyConexionBT.write(String.valueOf(Minutos.getText().subSequence(0,
1)));

MyConexionBT.write(String.valueOf(Minutos.getText().subSequence(1,
2)));

    }
else {
        Toast.makeText(getBaseContext(), "Los minutos o las horas no
son correctos", Toast.LENGTH_LONG).show();
        Hora.setText("");
        Minutos.setText("");
    }
```

Un cop enviada l'hora, es disposa tant a introduir els valors corresponents a l'apagat dels dos llums i de la calor, com al d'encesa.

L'hora d'apagada de les llums és diferent de l'hora de la calor, hi ha dues de diferència, per tant es passa el valor de l'hora d'apagada (String) a un valor(Integer) entre 0-2400 per poder sumar 200, tot seguit es comprova que aquest valor no sigui superior a 2400 i en el cas que ho sigui, es resta el 2400, i es converteix el valor final(Integer) en el format inicial(String), mitjançant la funció switch.

```
PrincipalApagado.setText(a + ":" + b);
SecundarioApagado.setText(a + ":" + b);
Calordorm = Integer.parseInt(c) + 200;

if (Calordorm > 2390) {
    Calordorm = Calordorm - 2400;
    switch (String.valueOf(Calordorm).length()) {
        case 0:
            CalorApagado.setText("00:00");
            break;
        case 1:
            CalorApagado.setText("00:0" + String.valueOf(Calordorm));
            break;
        case 2:
            CalorApagado.setText("00:" + String.valueOf(Calordorm));
            break;
        case 3:
            CalorApagado.setText("0" +
String.valueOf(Calordorm).substring(0, 1) + ":" +
String.valueOf(Calordorm).substring(1, 3));
            break;
```

```
 case 4:

CalorApagado.setText(String.valueOf(Calordorm).substring(0, 2) + ":" +
String.valueOf(Calordorm).substring(2, 4));
            break;
    }
} else {
    switch (String.valueOf(Calordorm).length()) {
        case 0:
            CalorApagado.setText("00:00");
            break;
        case 1:
            CalorApagado.setText("00:0" + String.valueOf(Calordorm));
            break;
        case 2:
            CalorApagado.setText("00:" + String.valueOf(Calordorm));
            break;
        case 3:
            CalorApagado.setText("0" +
String.valueOf(Calordorm).substring(0, 1) + ":" +
String.valueOf(Calordorm).substring(1, 3));
            break;
        case 4:

CalorApagado.setText(String.valueOf(Calordorm).substring(0, 2) + ":" +
String.valueOf(Calordorm).substring(2, 4));
            break;
    }
}
```

Per introduir el valor d'encesa, s'utilitza el mateix mètode anterior però en aquest cas en comptes de sumar 200, se suma 1100, ja que l'encesa onze hores després que s'apagui la primera llum.

```
Encender = Integer.parseInt(c) + 1100;
if (Encender > 2390) {
    Encender = Encender - 2400;
    switch (String.valueOf(Encender).length()) {
        case 0:
            Encendido.setText("00:00");
            break;
        case 1:
            Encendido.setText("00:0" + String.valueOf(Encender));
            break;

        case 2:
            Encendido.setText("00:" + String.valueOf(Encender));
            break;
        case 3:
            Encendido.setText("0" +
String.valueOf(Encender).substring(0, 1) + ":" +
String.valueOf(Encender).substring(1, 3));
            break;
        case 4:
            Encendido.setText(String.valueOf(Encender).substring(0, 2)
+ ":" + String.valueOf(Encender).substring(2, 4));
            break;
    }
```

```
} else {

 switch (String.valueOf(Encender).length()) {
        case 0:
            Encendido.setText("00:00");
            break;

        case 1:
            Encendido.setText("00:0" + String.valueOf(Encender));
            break;
        case 2:
            Encendido.setText("00:" + String.valueOf(Encender));
            break;
        case 3:
            Encendido.setText("0" +
String.valueOf(Encender).substring(0, 1) + ":" +
String.valueOf(Encender).substring(1, 3));
            break;
        case 4:
            Encendido.setText(String.valueOf(Encender).substring(0, 2)
+ ":" + String.valueOf(Encender).substring(2, 4));
            break;
    }
}
```

Finalment, l'ultima acció que hi ha és l'activació del ventilador, que també dona lloc al mode automàtic del control de calor.

```
Ventilador.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {MyConexionBT.write("V");

    }
});
```

OnResume

La següent funció que s'executa és l'onResume, on s'iniciarà la connexió Bluetooth.

Per començar s'ha d'inicialitzar la funció i obtenir les adreces MAC del Bluetooth del DispositivosBT.

```
@Override
public void onResume()
{
    super.onResume();
    Intent intent = getIntent();
    address =
intent.getStringExtra(DispositivosBT.EXTRA_DEVICE_ADDRESS);
    BluetoothDevice device = btAdapter.getRemoteDevice(address);
```

Un cop enllaçada l'adreça, es crea el Socket per a produir la comunicació, en el cas que no s'aconsegueixi saltarà un missatge d'error. En el cas que s'hagi pogut crear, s'obre el Socket.

```
try
{
    btSocket = createBluetoothSocket(device);
} catch (IOException e) {
    Toast.makeText(getBaseContext(), "La creacción del Socket fallo",
Toast.LENGTH_LONG).show();
}
try
{
    btSocket.connect();
} catch (IOException e) {
    try {
        btSocket.close();
    } catch (IOException e2) {}
}
```

Un cop obert el Socket, s'inicia la comunicació Bluetooth i s'envia el primer caràcter per intentar començar el flux de dades.

```
MyConexionBT = new ConnectedThread(btSocket);
MyConexionBT.start();
try {
    MyConexionBT.write("c");

}catch (Exception e){
    Toast.makeText(getBaseContext(), "La Conexión fallo",
Toast.LENGTH_LONG).show();
}
```

El caràcter "c" serà rebut al microcontrolador i amb ell es començarà a enviar la informació sobre els mòduls connectats, que s'explicarà amb més detall en següent apartat.

OnStart

Un cop s'hagin executat les funcions onCreate i onResum, respectivament, ho fa l'onStart. Principalment dins d'aquesta funció es tracten els valors rebuts i se'ls hi dóna forma i sentit. Un cop tractats, s'envia al microcontrolador un valor per a determinar que s'han rebut les dades.

Primer es crea la funció i tot seguit es defineix l'acció Handler, el qual té la funció d'enviar i processar els objectes Message. En el moment que hi ha dades al buffer, es llegeix el missatge per començar a processar les dades.

```
@Override
public void onStart(){
  super.onStart();

  bluetoothIn = new Handler() {
    public void handleMessage(android.os.Message msg) {
      if (msg.what == handlerState) {
        String readMessage = (String) msg.obj;
        DataStringIN.append(readMessage);
```

Un cop llegit el missatge és el moment de processar-lo, per fer-ho primer es llegeix el últim valor i mitjançant l'acció "switch" es comença a processar.

```
String Select=DataStringIN.substring(DataStringIN.length()-
1,DataStringIN.length());

    switch (Select){}
```

Es poden donar dos casos, que arribi un # o que arribi $, en el primer cas el que es farà és la comprovació que la llargada de les dades no sigui diferent de 9, si es compleix, donarà lloc al processament de la dada.

```
case "#":
    if (DataStringIN.length()==9) {}
```

Primer se separa els diferents valors de la dada rebuda. El dos primers són la temperatura i humitat, el tercer és l'estat dels relés i l'últim és l'estat del ventilador, que també marca si el control de la temperatura està en automàtic.

```
String dataInPrint = DataStringIN.substring(DataStringIN.length() - 4, DataStringIN.length() -
2);//Temperatura
String dataInPrint3 = DataStringIN.substring(DataStringIN.length() - 6, DataStringIN.length() -
4);//Humedad
String dataInPrint4 = DataStringIN.substring(0, DataStringIN.length() - 6); //Estado de luzes
String dataInPrint2 = DataStringIN.substring(DataStringIN.length() - 2, DataStringIN.length() -
1);//Ventilador
```

Un cop separats, s'enllacen els valors de temperatura i humitat. Si hi ha varietat en els valors anteriors, es produirà un moviment en la barra que marca la temperatura, humitat o ambdós.

```
Temp.setText("<-"          +          dataInPrint          +          "°C");
MovimientoTemp                  =                  (7.575                  *
Double.parseDouble(String.valueOf(dataInPrint)));

Hum.setText("<-"          +          dataInPrint3          +          "%");
MovimientoHum                  =                  (3.03                  *
Double.parseDouble(String.valueOf(dataInPrint3)));

TemperaturaBar.setProgress(Integer.parseInt(String.valueOf(dataInPrint
)));
HumedadBar.setProgress(Integer.parseInt(dataInPrint3));


Temp.setTranslationY(-1*(int)MovimientoTemp);
Hum.setTranslationY(-1*(int)MovimientoHum);
```

Per tractar el valor de l'estat del ventilador s'utilitza un switch, però només té un cas, que el valor sigui 1. Si es compleix, la imatge del ventilador entrarà en moviment rotatiu.[1]

```
switch (dataInPrint2){
    case "1":
        new CountDownTimer(5000, 50) {
        public void onTick(long millisUntilFinished) {
            Cont = Cont + 2;
            Ventilador.setRotation(Cont);
        }

        public void onFinish() {

        }
        }.start();
        break;
}
```

El següent valor a tractar és l'estat dels relés. El valor rebut serà entre "000" fins a "007", el qual engloba tots els possibles estats dels relés. Aquest valor prové d'una suma explicada en el codi d'Arduino més endavant.

```
switch (dataInPrint4) {
    case "000":
        Calor.setChecked(FALSE);
        Luz.setChecked(FALSE);
        Luz2.setChecked(FALSE);
```

[1] Es podria utilitzar la funció "if" però no es produïa el moviment rotatiu del ventilador desitjat.

```
            break;
        case "001":
            Calor.setChecked(TRUE);
            Luz.setChecked(FALSE);
            Luz2.setChecked(FALSE);
            break;
        case "002":
            Calor.setChecked(FALSE);
            Luz.setChecked(TRUE);
            Luz2.setChecked(FALSE);
            break;
        case "003":
            Calor.setChecked(TRUE);
            Luz.setChecked(TRUE);
            Luz2.setChecked(FALSE);
            break;
        case "004":
            Calor.setChecked(FALSE);
            Luz.setChecked(FALSE);
            Luz2.setChecked(TRUE);
            break;
        case "005":
            Calor.setChecked(TRUE);
            Luz.setChecked(FALSE);
            Luz2.setChecked(TRUE);
            break;
        case "006":
            Calor.setChecked(FALSE);
            Luz.setChecked(TRUE);
            Luz2.setChecked(TRUE);
            break;
        case "007":
            Calor.setChecked(TRUE);
            Luz.setChecked(TRUE);
            Luz2.setChecked(TRUE);
            break;

}
```

Un cop processades totes les dades, s'esborra el buffer i s'envia mitjançant la funció write, que és una subrutina que s'explica més endavant, un valor a l'Arduino amb la funció d'"avisar" que la dada ha sigut ben rebuda i pot tornar a enviar un altre.

```
DataStringIN.delete(0, DataStringIN.length());
try {
    MyConexionBT.write("T");
}catch (Exception e){
    Toast.makeText(getBaseContext(), "La Conexión fallo",
Toast.LENGTH_LONG).show();
}
```

Un cop finalitzat el primer dels casos, es dóna pas al següent, el cas que el valor final de la dada rebuda sigui $, que correspon a la inicialització de la comunicació entre el microcontrolador i el mòbil.

En rebre aquesta dada final, s'inicialitzen les hores d'encesa i apagat dels relés, mitjançant el mateix mètode explicat en l'apartat onCreate. També s'activa la confirmació que hi ha comunicació entre l'aparell mòbil i el microcontrolador.

```
case "$":
    String dataInPrint5 = DataStringIN.substring(0,
DataStringIN.length() - 1);

    Comp.setChecked(TRUE);
    Comp.setText("Conectado");

PrincipalApagado.setText(dataInPrint5.substring(0,2)+":"+dataInPrint5.
substring(2,4));

SecundarioApagado.setText(dataInPrint5.substring(0,2)+":"+dataInPrint5
.substring(2,4));
    Calordorm=Integer.parseInt(dataInPrint5)+200;
    Encender=Integer.parseInt(dataInPrint5)+1100;
```

Un cop processades les hores, és borra el buffer i s'envia la dada per a què es comenci a rebre els valors del cas #.

```
DataStringIN.delete(0, DataStringIN.length());
try {
    MyConexionBT.write("T");
}catch (Exception e){
    Toast.makeText(getBaseContext(), "La Conexión fallo",
Toast.LENGTH_LONG).show();
}
```

Finalment es verifica l'estat de la connexió bluetooth.

```
btAdapter = BluetoothAdapter.getDefaultAdapter();
VerificarEstadoBT();
```

Altres

En aquest apartat s'acabarà d'explicar la resta de funcions que falten per finalitzar l'explicació del programa d'Android Studio.

La primera funció és l'onPause. Al finalitzar la pantalla actual, entrarà en acció aquesta funció i el que farà és finalitzar la comunicació, és a dir, tancarà el Socket.

```
@Override
public void onPause()
{
    super.onPause();
   try{ // Cuando se sale de la aplicación esta parte permite
        // que no se deje abierto el socket
       btSocket.close();
       finish();
   } catch (IOException e2) {}
}
```

De la mateixa manera que en la pantalla de DispostivosBT, en l'actual també existeix una funció per verificar que el Bluetooth estigui disponible i que s'activi en el cas que no ho estigui.

```
public void VerificarEstadoBT() {

    if(btAdapter==null) {
        Toast.makeText(getBaseContext(), "El dispositivo no soporta
bluetooth", Toast.LENGTH_LONG).show();
    } else {
        if (btAdapter.isEnabled()) {
        } else {
            Intent enableBtIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
            startActivityForResult(enableBtIntent, 1);
        }
    }
}
```

Per finalitzar, l'última funció creada és aquella que permet la creació de l'esdeveniment de comunicació.

```
private class ConnectedThread extends Thread
{
    private final InputStream mmInStream;
    private final OutputStream mmOutStream;
```

```
    public ConnectedThread(BluetoothSocket socket)
    {
        InputStream tmpIn = null;
        OutputStream tmpOut = null;
        try
        {
            tmpIn = socket.getInputStream();
            tmpOut = socket.getOutputStream();
        } catch (IOException e) { }
        mmInStream = tmpIn;
        mmOutStream = tmpOut;
    }
```

Dins de la funció ConnectedThread, i mitjançant les dues variables creades, la In i la Out, es creen dues funcions per llegir els valors emmagatzemats al buffer i per enviar els caràcters.

```
public void run()
{
    byte[] buffer = new byte[256];
    int bytes;

    // Se mantiene en modo escucha para determinar el ingreso de datos
    while (true) {
        try {
            bytes = mmInStream.read(buffer);
            String readMessage = new String(buffer, 0, bytes);
            // Envia los datos obtenidos hacia el evento via handler
            bluetoothIn.obtainMessage(handlerState, bytes, -1,
readMessage).sendToTarget();

        } catch (IOException e) {
            Toast.makeText(getBaseContext(), "La Conexión fallo",
Toast.LENGTH_LONG).show();
            break;
        }
    }
}
//Envio de trama
public void write(String input)
{
    try {
        mmOutStream.write(input.getBytes());
    }
    catch (IOException e)
    {
        Toast.makeText(getBaseContext(), "La Conexión fallo",
Toast.LENGTH_LONG).show();
        finish();
    }
}
```

8.- Explicació codi Arduino

Aquest apartat serveix per entendre la lògica de programació, que s'ha dut a terme en aquest projecte, aplicada al microcontrolador Arduino i als mòduls connectats.

A diferència de l'Android Studio, en aquest programa només hi ha un fitxer de programació.[2]

El diagrama de blocs del qual parteix el programa és el següent:

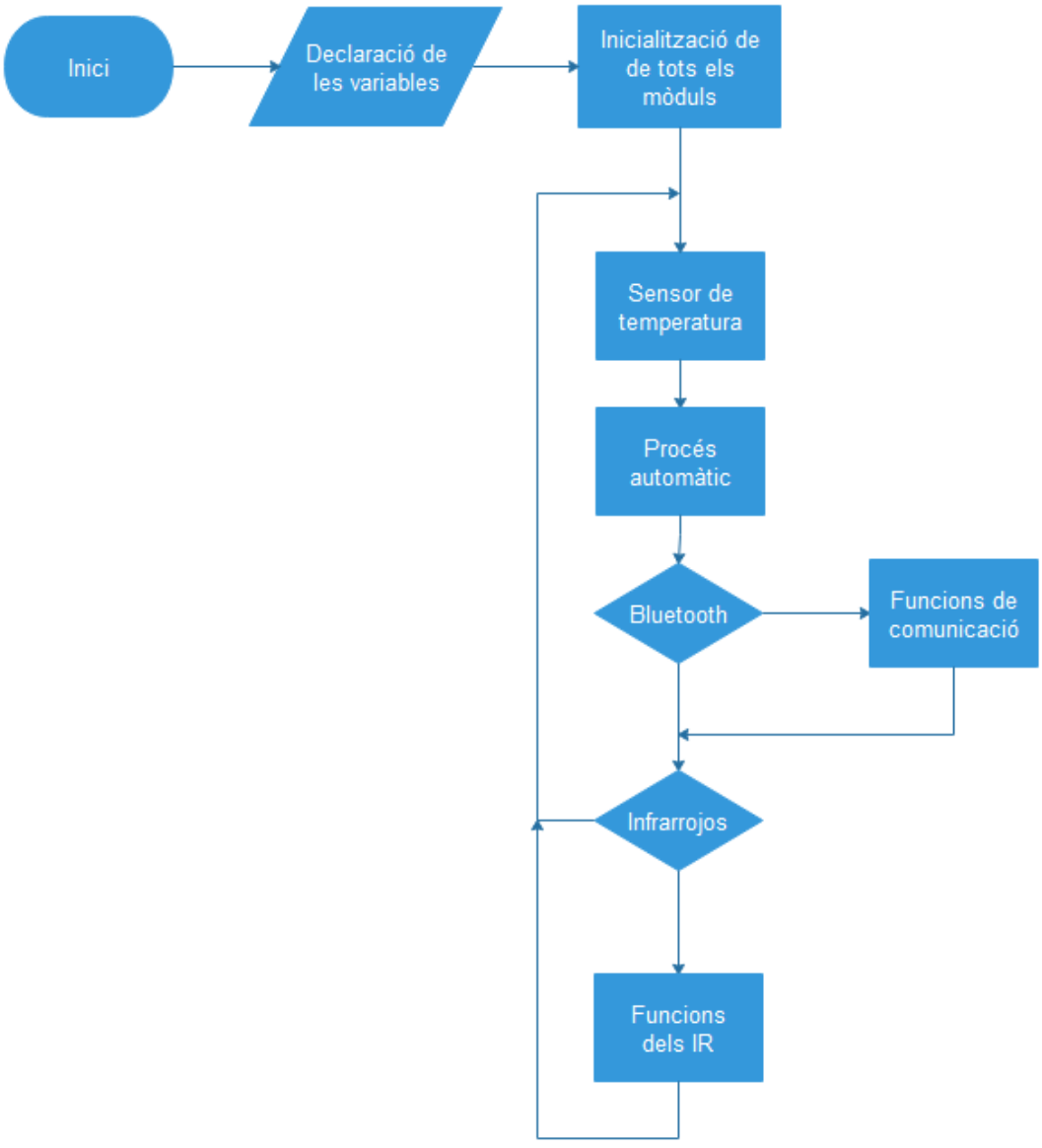


Figura 8.1.- Diagrama de l'arduino.

[2] Existeixen les llibreries incorporades al fixer.

Seguint el diagrama de blocs anterior, primerament es declaren les llibreries i totes les variables que són utilitzades al llarg del programa.

```
#include <IRremote.h>
#include <IRremoteInt.h>
#include <SimpleDHT.h>
#include <SoftwareSerial.h>
#include <LiquidCrystal.h>
#include <Wire.h>
#include "RTClib.h"
#define TIMER_ENABLE_INTR    (TIMSK5 = _BV(OCIE5A))
#define TIMER_DISABLE_INTR   (TIMSK5 = 0)

LiquidCrystal lcd(43, 41, 39, 37, 35, 33);//(7, 8, 9, 10, 11, 12)
RTC_DS3231 rtc;
String Recive = "";
int Ventilador=4;
int Pantalla=5;
bool Inicio=false;
int Temp = 2; //Pin del sensor de temperatura
int Reset=12;
SimpleDHT11 dht11;
int Calor=52;
int Luz1=50;
int Luz2=48;
int EstadoLuces;
int temperatureCheck = 0;
int humidityCheck = 0;
byte temperature = 0;
byte humidity = 0;
String LCD,Fecha,setFecha,DatosRele,Tempp,Hora,setHoraDesp;
String setLuzDorm="2100";
String setLuz1Dorm="2100";
String setCalorDorm2="2300";
String setDesp2="0800";
unsigned int setDesp= 800;
unsigned int setCalorDorm=2300;
bool EstadoLCD=false;
bool inicio=false;
int Receptor = 3;
String Minutos,Horas;
IRrecv REC(Receptor);
decode_results RES;
```

Un cop declarades i iniciades les variables, apareix la funció setup, la qual només s'executa una vegada. En aquesta funció és on s'inicialitzen tots els mòduls que hi ha connectats a l'Arduino.

```
void setup() {}
```

Primer s'inicia el port de comunicació, en aquest cas, hi ha dos, el Serial i el Serial1. El primer és utilitzat per fer proves i el segon és el que correspon al mòdul Bluetooth. S'envia una dada, per si hi ha algun dispositiu esperant la connexió.

```
Serial.begin(9600);//Baudios
Serial1.begin(9600);
Serial1.print("Modulo Conectado$");
```

Seguidament s'inicien el mòdul d'infrarojos, el LCD i una pausa de tres segons per a què finalitzi la inicialització del microcontrolador.

```
REC.enableIRIn();
lcd.begin(16, 2);
lcd.print("App Camaleon");
delay(3000);
```

El pròxim mòdul a iniciar-se és el rellotge, el qual si en algun moment perd la tensió, s'ajustarà gràcies a la pila interna que té.

```
if (! rtc.begin()) {while (1);}
 if (rtc.lostPower()) { rtc.adjust(DateTime(F(__DATE__), F(__TIME__))); }
```

Finalment, per acabar la funció setup, s'inicien tots els pins de control.

```
pinMode(Calor,OUTPUT);
pinMode(Luz1,OUTPUT)
pinMode(Luz2,OUTPUT);
pinMode(Reset,OUTPUT);
pinMode(Ventilador,OUTPUT);
pinMode(Pantalla,OUTPUT);

digitalWrite(Calor,HIGH);
digitalWrite(Luz1,HIGH);
digitalWrite(Luz2,LOW);
digitalWrite(Ventilador,HIGH);
digitalWrite(Pantalla,HIGH);
```

La següent funció és la loop. S'executa contínuament en finalitzar tots els processos i accions que hi ha dins seu.

```
void loop() { }
```

El primer procés que hi ha és l'actualització del valor del rellotge i el correcte format (XX:XX) per poder treballar amb ell posteriorment.

```
DateTime now = rtc.now();
if(now.minute()<10){Minutos="0"+(String)now.minute();}else{Minutos=(String)now.minute();
if(now.hour()<10){Horas="0"+(String)now.hour();}else{Horas=(String)now.hour();}
Fecha = (String)now.day() + "/" + (String)now.month() + "      " + Horas + ":" + Minutos + "   ";
setFecha=Horas+Minutos;
```

Un cop adquirit la data i l'hora, es llegeix els valors de temperatura i humitat del sensor. Per poder llegir-ho és necessari desactivar una interrupció creada per la llibreria del sensor d'infrarojos.

```
TIMER_DISABLE_INTR ;
dht11.read(Temp, &temperature, &humidity, NULL);
TIMER_ENABLE_INTR ;
delay(300);
```

El codi que hi ha a continuació correspon al control de la calor. En el cas que la temperatura sigui major que 30, la calor i el ventilador estiguin encesos, es farà un canvi de la calor per la llum secundària.

En el cas que la temperatura sigui inferior a 29, la llum secundària i el ventilador estiguin encesos, es farà el canvi de la llum secundària a la calor.

```
 if(temperature > 30 && digitalRead(Calor)==HIGH && digitalRead(Ventilador)==HIGH){

   digitalWrite(Calor,LOW);
   digitalWrite(Luz1,HIGH);
   digitalWrite(Luz2,HIGH);
  }

  if(temperature < 29 && digitalRead(Calor)==LOW && digitalRead(Ventilador)==HIGH){

   digitalWrite(Calor,HIGH);
   digitalWrite(Luz1,HIGH);
   digitalWrite(Luz2,LOW);
   }
```

Tot seguit es procedeix a mostra per la pantalla LCD, el dia, l'hora, la temperatura i la humitat.

```
LCD= "Temp: " + (String)temperature+" "+"Hum: " + (String)humidity;
lcd.setCursor(0, 0);
lcd.print(Fecha);
lcd.setCursor(0, 1);
lcd.print(LCD);
```

El codi següent correspon a la comunicació. Si es rep algun caràcter, aquesta funció s'activarà i es llegirà el valor que hi hagi al buffer.

Depenent del valor, mitjançant un switch, s'executaran unes accions o unes altres.

```
if (Serial1.available())
 {
 char Rec;
 Rec = Serial1.read();
 Serial.print(Rec);
      switch (Rec) {}
```

En el cas que el caràcter sigui "T", s'enviarà el valor de l'estat dels relés, la temperatura i la humitat.

```
case 'T':

          delay(100);
          if(digitalRead(Calor)==HIGH){EstadoLuces= EstadoLuces + 1; }
          if(digitalRead(Luz1)==HIGH) {EstadoLuces= EstadoLuces + 2; }
         if(digitalRead(Luz2)==HIGH) {EstadoLuces= EstadoLuces + 4;}
         DatosRele="00"+(String)EstadoLuces;
         Tempp=DatosRele+(String)humidity+(String)temperature +
         (String)digitalRead(Ventilador)+ "#";
         Serial1.print(Tempp);
         EstadoLuces=0;

break;
```

El cas “c”és el valor d’inicialització de la comunicació, en el qual s’envia el valor d’encesa i apagat de relés.

```
case 'c':
       delay(100);
       Serial1.print(setLuzDorm+"$");
break;
```

Els casos "Q", "W", "R" i "V" tenen la funció de canviar l'estat de la sortida del microcontrolador en qüestió. Aquestes sortides són: la calor, la llum principal, la llum secundària i el ventilador, respectivament.

```
case 'Q':
        if(digitalRead(Calor)==HIGH){digitalWrite(Calor,LOW);}
        else{digitalWrite(Calor,HIGH);}
break;

case 'W':
          if(digitalRead(Luz1)==HIGH){digitalWrite(Luz1,LOW);}
          else{digitalWrite(Luz1,HIGH);}

break;

case 'R':
          if(digitalRead(Luz2)==HIGH){digitalWrite(Luz2,LOW);}
          else{digitalWrite(Luz2,HIGH);}

break;

case 'V':
        if(digitalRead(Ventilador)==HIGH){digitalWrite(Ventilador,LOW);}
        else{digitalWrite(Ventilador,HIGH);}

break;
```

Els següents casos són número del 0-9, on l'única funció que hi ha és la de sumar el valor corresponent a una variable. Aquí només apareixen tres casos, els altres es poden comprovar a l'Annex II.

```
case'0':
       Hora=Hora+"0";
break;

case'1':
       Hora=Hora+"1" ;
break;

case'2':
        Hora=Hora+"2";
break;
```

Un cop es completa la variable "Hora", es duu a terme l'acció d'establir les hores d'apagada i d'encesa. També transforma el valor en el format correcte (HH:MM), igual que al principi de la funció setup, per a poder ser tractat posteriorment.

```
if(Hora.length()==4){
      setLuzDorm=Hora;
      setLuz1Dorm=Hora;
      setCalorDorm=setLuzDorm.toInt() + 200;
      setDesp= setLuzDorm.toInt() + 1100;

      if (setDesp > 2359){
          setDesp=setDesp-2400;

          if(String(setDesp).length()==3){setDesp2="0"+(String)setDesp;}
          else{setDesp2=(String)setDesp;}

      }else {setDesp2=(String)setDesp;}

     if (setCalorDorm > 2359){
        setCalorDorm=setCalorDorm-2400;

        switch(String(setCalorDorm).length())

        {
        case 4:
          setCalorDorm2=(String)setCalorDorm;
        break;
       case 3:
          setCalorDorm2="0"+(String)setCalorDorm;
       break;
       case 2:
          setCalorDorm2="00"+(String)setCalorDorm;
       break;
       case 1:
         setCalorDorm2="000"+(String)setCalorDorm;
       break;
       case 0:
         setCalorDorm2="0000";
       break;
     }
```

```
}else{
if(String(setCalorDorm).length()==3){setCalorDorm2="0"+(String)setCalorDorm;}
          else{setCalorDorm2=(String)setCalorDorm;}

            }
     Hora="";
  }
```

Es fa una comparació entre l'hora establerta per l'usuari i l'hora actual, en el moment que coincideixi i que els relés estiguin en l'estat oposat, donarà lloc a fer el canvi d'estat, és a dir, quan arribi a l'hora establerta, s'apagaran les llums

En el cas que estiguessin oberts, i l'hora coincideixi amb l'hora d'encesa, s'encendran tots tres i el ventilador, en el cas que estiguin apagats.

```
If (setLuzDorm.substring(0,2)==setFecha.substring(0,2) &&
setLuzDorm.substring(2,4)=                                    =setFecha.substring(2,4)
&&( digitalRead(Luz1)==HIGH || digitalRead(Luz2)==HIGH) )
{
   digitalWrite(Luz1,LOW);
   digitalWrite(Luz2,LOW);
}

if(setCalorDorm2.substring(0,2)==setFecha.substring(0,2) &&
setCalorDorm2.substring(2,4)= =setFecha.substring(2,4) &&
digitalRead(Calor)==HIGH )
 {
    digitalWrite(Calor,LOW);
    digitalWrite(Ventilador,LOW);
    digitalWrite(Pantalla,LOW);
}

if(setDesp2.substring(0,2)==setFecha.substring(0,2) && setDesp2.substring(2,4)=
=setFecha.substring(2,4)&&(digitalRead(Calor)==LOW || digitalRead(Luz1)==LOW) )
{
   digitalWrite(Calor,HIGH);
   digitalWrite(Luz1,HIGH);
   digitalWrite(Ventilador,HIGH);
   digitalWrite(Pantalla,HIGH);
}
```

Per acabar el codi d'aquest programa, falta l'explicació del sensor d'infrarojos. En el moment que es polsi algun botó del comandament, s'iniciarà l'acció de seleccionar, segons el botó, l'acció a realitzar.

Cada u té una acció diferent, que és explicada en l'apartat 2

A continuació es mostrarà el codi de totes les funcions usades.[3]

```
if (REC.decode(&RES))
{
  switch(RES.value)
  {
   case 0xFFA25D: //POWER
          if (EstadoLCD==true){
             lcd.noDisplay();
             EstadoLCD=false;
             digitalWrite(Pantalla,LOW);
          }else{
            lcd.display();
            EstadoLCD=true;
            digitalWrite(Pantalla,HIGH);
          }
  break;

  case 0xFF02FD: //Pause
           if(digitalRead(Ventilador)==HIGH){digitalWrite(Ventilador,LOW);}
          else{digitalWrite(Ventilador,HIGH);}
  break;
  case 0xFF9867:  //EQ
          lcd.setCursor(0, 1);
         LCD="Encendido: "+setDesp2.substring(0,2)+":"+setDesp2.substring(2,4);
          lcd.print(LCD);
         delay(1000);

  break;

  case 0xFF6897: //0
         digitalWrite(Calor,HIGH);
        digitalWrite(Luz1,HIGH);
        digitalWrite(Luz2,LOW);
        setDesp2= "0800";
        setCalorDorm2="2300";
```

[3] En el codi original apareixen tots els botons encara que no s'utilitzin, per si en un futur fossin necessaris.

```
        setLuzDorm="2100";
        setLuz1Dorm="2100";
        Serial1.print(setLuzDorm+"$");

  break;

 case 0xFF30CF: //1
        if(digitalRead(Calor)==HIGH){digitalWrite(Calor,LOW); }
        else{digitalWrite(Calor,HIGH);}

  break;

 case 0xFF18E7: //2
         if(digitalRead(Luz1)==HIGH){digitalWrite(Luz1,LOW);}
        else{digitalWrite(Luz1,HIGH);}
 break;

case 0xFF7A85: //3
           if(digitalRead(Luz2)==HIGH){digitalWrite(Luz2,LOW);}
           else{digitalWrite(Luz2,HIGH);}
break;

case 0xFF10EF: //4
            lcd.setCursor(0, 1);
            LCD="Es: "+(String)digitalRead(Calor)+"   H:
"+setCalorDorm2.substring(0,2)  +":"
            +  setCalorDorm2.substring(2,4);
           lcd.print(LCD);
           delay(1000);
break;

case 0xFF38C7: //5
          lcd.setCursor(0, 1);
          LCD="Es:"+(String)digitalRead(Luz1)+" H:
"+String(setLuzDorm).substring(0,2) +":"
          +setLuzDorm.substring(2,4);
          lcd.print(LCD);
          delay(1000);

break;

case 0xFF5AA5: //6
        lcd.setCursor(0, 1);
        LCD="Es: "+(String)digitalRead(Luz2)+"   H: " + setLuz1Dorm.substring(0,2)+
":" +
        setLuz1Dorm.substring(2,4);
```

```
        lcd.print(LCD);
        delay(1000);
break;

default:
         lcd.setCursor(4, 0);
         lcd.print("XXXX");

}
delay(100);
REC.resume();

}
```

I d'aquesta manera finalitza l'explicació del codi d'Arduino. El codi complet està disponible a l'Annex II.

9.- Conclusions

Finalment s'ha arribat a l'objectiu inicial, la creació d'un sistema capaç de controlar les funcions bàsiques del terrari d'un camaleó. La creació d'aquest sistema de control soluciona el problema d'haver d'estar pendent de les hores de llum que es proporciona a l'animal i de la incertesa de la quantitat de calor a l'interior del terrari.

Per fer-ho s'ha creat un sistema de control, el qual és governat per una placa de desenvolupament hardware i incorpora uns relés per fer el control de la il·luminació, un rellotge per dur a terme l'automatització, un sensor de temperatura. Aquest sistema pot ser controlat per una aplicació mòbil o per un comandament a distància.

S'ha intentat fer un canvi de comunicació, a Ethernet, per així poder tenir un major rang de control del terrari, però finalment no s'ha vist viable. També s'ha intentat la incorporació d'una càmera, però el fet d'incorporar-la feia molt lent tot el sistema, i amb aquest tipus de comunicació, Bluetooth, es considera inviable, ja que el rang d'actuació és petit, per tant implica estar a prop del terrari.

L'ús que s'ha fet de l'aplicació és més de caràcter industrial que comercial, utilitzant els coneixements adquirits al llarg del grau. Per exemple la creació de sistemes de control i adquisició de dades prové de Sistemes d'Informació i Comunicació Industrial, el funcionament d'un microcontrolador d'Informàtica Industrial, la creació de circuits electrònics de diverses assignatures diferents (EAEIA, TEEIA, EPEIA...) i la idea de la creació d'un sistema controlat d'Equips Electrònics. Aquest projecte ha servit per adquirir un nou llenguatge de programació, Java, per unir conceptes adquirits, per cercar i seleccionar recursos adequats pel funcionament correcte del sistema, entre altres coses.

Com conclusió final es pot dir que s'ha complert l'objectiu de crear un sistema capaç de resoldre la problemàtica de la necessitat de regular l'horari d'il•luminació i calor d'un terrari.

9.1.-Futures línies de treball

Si es continués el treball, es podria fer un canvi de comunicació, per exemple Ethernet, ja que suposaria ampliar el rang d'acció sobre el sistema ja creat. Al tenir aquest tipus de comunicació es podria incorporar una càmera IP, per tenir visualització real del terrari i així poder comprovar que tot funciona correctament. En el cas de crear la comunicació per Ethernet, s'hauria de millorar l'aplicació del mòbil, per poder incorporar la visualització de la càmera.

Es podria incorporar un sensor de llum per detectar errors, i en el cas que existissin, avisar mitjançant l'aplicació de mòbil o hi hagués un registre d'errors i un cop encesa l'aplicació es mostressin. També es podria millorar la part de visual del projecte, el hardware i fer un sistema fix i una instal·lació més petita dels tres portalàmpades.

Com a última idea d'ampliació podria ser la incorporació d'un humidificador per al moment que la humitat sigui menor a la desitjada, humidifiqués el terrari.

Bibliografia

[1] Chamaeleo calyptratus. Wikipedia. https://es.wikipedia.org/wiki/Chamaeleo_calyptratus

[2] Reptiles y anfibios. Decogarden. https://www.hogarmania.com/mascotas/otras/reptiles-anfibios/200903/camaleon-yemen-chamaeleo-calyptratus-6532.html

[3] Arduino. Vikipèdia. https://ca.wikipedia.org/wiki/Arduino

[4] ESQUEMA DE PATILLAJE (PINOUT) DE ARDUINO UNO, NANO, MINI Y MEGA. Luis Llamas.28/10/2015. https://www.luisllamas.es/esquema-de-patillaje-de-arduino-pinout/

[5] MÓDULO BLUETOOTH HC-06. Prometec. https://www.prometec.net/bt-hc06/

[6] Comunicación Bluetooth. Innova Domotics. http://www.innovadomotics.com/mn-tuto/mn-android/proyectos/23-and-cnm-bt.html?start=5

[7] Imatge HC-06 i Connexió. Buscador Google. https://www.google.es/search?biw=1920&bih=920&tbm=isch&sa=1&ei=Ku_XXNvWI7yFjLsPzICV6As&q=hc-06&oq=hc-06&gs_l=img.3..35i39j0i19l9.10714.10714..10955...0.0..0.73.73.1.....1....1..gws-wiz-img.8IEyWH7J86Y

[8] Imatge Relés. Buscador Google. https://www.google.es/search?q=hl-54s+v1.0+datasheet&source=lnms&tbm=isch&sa=X&ved=0ahUKEwjapKbW5pjiAhUsx4UKHduWCtgQ_AUIDigB&biw=1920&bih=920#imgrc=j0xl_xFzQ8a5BM:

[9] Connexió Relés.Buscador Google. https://www.google.es/search?q=hl-54s+v1.0+datasheet&source=lnms&tbm=isch&sa=X&ved=0ahUKEwjapKbW5pjiAhUsx4UKHduWCtgQ_AUIDigB&biw=1920&bih=969#imgrc=wcc54B8WhapjmM:

[10] Imatge AX-1838HS. https://www.google.es/search?q=ax-1838hs+datasheet&source=lnms&tbm=isch&sa=X&ved=0ahUKEwiUgJD

Z7ZjiAhWxzoUKHXS9AucQ_AUIDigB&biw=1920&bih=920#imgrc=mlO-qTJtiOafcM:

[11] "Hello World!".Arduino. https://www.arduino.cc/en/Tutorial/HelloWorld

[12] RELOJ Y CALENDARIO EN ARDUINO CON LOS RTC DS1307 Y DS3231. Luis Llamas. 18/10/2016. https://www.luisllamas.es/reloj-y-calendario-en-arduino-con-los-rtc-ds1307-y-ds3231/

[13] Transistor de unión bipolar.Wikipedia. https://es.wikipedia.org/wiki/Transistor_de_uni%C3%B3n_bipolar#Tipos_de_Transistor_de_Uni%C3%B3n_Bipolar

[14] PN2222. FairChild. https://www.onsemi.com/pub/Collateral/PN2222-D.PDF

[15] Resistencia eléctrica. Wikipedia. https://es.wikipedia.org/wiki/Resistencia_el%C3%A9ctrica

[16] Imatge Ventilador.Buscador Google. https://www.google.com/search?q=ventilador+5+v+dc+datasheet&rlz=1C1CHBD_esES787ES787&tbm=isch&source=iu&ictx=1&fir=a2G9Wlu23uZSaM%253A%252CYUclZ0uUPf8zSM%252C_&vet=1&usg=AI4_-kQO9c8Qph1jRgvwfqqClI3foDPj1w&sa=X&ved=2ahUKEwi34YCL_KfiAhUwxoUKHU6HDokQ9QEwAnoECAkQBA#imgrc=a2G9Wlu23uZSaM:

[17] Imatge Portalàmpades. Buscador Google. https://www.google.com/search?rlz=1C1CHBD_esES787ES787&tbm=isch&sa=1&ei=T37hXI_zLYCE1fAPkOyK8AM&q=portalamparas+camaleon&oq=portalam&gs_l=img.1.0.35i39j0l6j0i30l3.2258.3320..5048...0.0..0.75.533.8.....1....1..gws-wiz-img.....0i67.cpCrUCRtAH4#imgrc=ciJOWQvmsiyEsM:

[18] Logotip Android. Buscador Google. https://www.google.com/search?q=logo+android+studio&rlz=1C1CHBD_esES787ES787&source=lnms&tbm=isch&sa=X&ved=0ahUKEwjb5sP1ibLiAhXm0eAKHf9eDxoQ_AUIDigB&biw=957&bih=910

[19] Android Studio. Wikipedia.
https://es.wikipedia.org/wiki/Android_Studio

[20] El ciclo de vida del Activity en Android. Erick Tijero Urbano. 23/07/2016. https://stories.devacademy.la/android-ciclo-de-vida-de-un-activity-b2a68c506961

[21] Logotip Arduino.Buscador Google.
https://www.google.com/search?rlz=1C1CHBD_esES787ES787&biw=1920&bih=969&tbm=isch&sa=1&ei=3gjoXKKWYM8HjgweS9qWADw&q=logotip+arduino+ide&oq=logotip+arduino+ide&gs_l=img.3...1087.1087..1968...0.0..0.75.75.1.....0....1..gws-wiz-img.Wz-YVyTLeCk

[22]Crear un en Android Studio. Andrea Ardións. 06/04/2015.
https://androidstudiofaqs.com/tutoriales/crear-un-layout-en-android-studio

[23] Cómo poner el botón de atrás en Android Studio. Andrea Ardións. 01/10/2015. https://androidstudiofaqs.com/tutoriales/como-poner-el-boton-de-atras-en-android-studio

[24] Popup Menu - Android Studio Tutorial. Coding in flow. 28/10/2017.
https://www.youtube.com/watch?v=s1fW7CpiB9c

[25] IF … ELSE. Home and Learn.
https://www.homeandlearn.co.uk/java/java_if_else_statements.html

[26] String length(). Arduino.
https://www.arduino.cc/en/Tutorial/StringLength

[27] Datos Sectoriales. ANFAAC. Disponible en:
http://www.anfaac.org/datos-sectoriales/

[28] Federico Guede Fernàndez, C.J. La implementación de una aplicación móvil que detecta somnolencia en la conducción. Projecte Final de Carrera, Universidad Politécnica de Catalunya, 2012.

[29] Eduard Ballester i Robert Piqué. Electrónica de Potència. 2011

Annex I: Codi Android Studio

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.tfg.appcamaleon">
    <!-- To auto-complete the email text field in the login form
with the user's emails -->
    <uses-permission
android:name="android.permission.GET_ACCOUNTS" />
    <uses-permission
android:name="android.permission.READ_PROFILE" />
    <uses-permission
android:name="android.permission.READ_CONTACTS" />
    <uses-permission android:name="android.permission.BLUETOOTH"
/>
    <uses-permission
android:name="android.permission.BLUETOOTH_ADMIN" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="App_Camaleon"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".Portada"
            android:screenOrientation="portrait">
            <intent-filter>
                <action
android:name="android.intent.action.MAIN" />

                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".DispositivosBT"
            android:screenOrientation="portrait" />
        <activity
            android:name=".Principal"
            android:screenOrientation="portrait" />
    </application>

</manifest>
```

Portada.java

```
package com.tfg.appcamaleon;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.MenuItem;
import android.view.View;
import android.widget.ImageView;
import android.os.CountDownTimer;
import android.widget.PopupMenu;

public class Portada extends AppCompatActivity {
    ImageView Camaleon;
    double cont = 0;
    double Cont2 = 0, angulo = 0;
    int Valor=0;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_portada);
        Camaleon = (ImageView) findViewById(R.id.idFotoCamaleon);

        new CountDownTimer(2600, 100) {

            public void onTick(long millisUntilFinished) {
                cont--;
                Cont2 = Cont2 - 0.61538;
                Camaleon.setTranslationX((float) cont);
                Camaleon.setTranslationY((float) Cont2);
            }

            public void onFinish() {
                cont = Camaleon.getTranslationX();
                Cont2 = Camaleon.getTranslationY();
                angulo = Camaleon.getRotation();
                new CountDownTimer(8200, 100) {
                    public void onTick(long millisUntilFinished) {
                        cont--;
                        angulo = angulo - 0.244;
                        Cont2 = Cont2 - 0.36;
                        Camaleon.setTranslationX((float) cont);
                        Camaleon.setTranslationY((float) Cont2);
                        Camaleon.setRotation((float) angulo);
                    }

                    public void onFinish() {
                        cont = Camaleon.getTranslationX();
                        Cont2 = Camaleon.getTranslationY();
                        angulo = Camaleon.getRotation();
                        new CountDownTimer(20000, 50) {
                      public void onTick(long millisUntilFinished) {
                            cont--;
                            Cont2 = Cont2 - 0.22;
                            Camaleon.setTranslationX((float) cont);
                            Camaleon.setTranslationY((float) Cont2);
                             }
```

```
   public void onFinish() {
        cont = Camaleon.getTranslationX();
        Cont2 = Camaleon.getTranslationY();
        new CountDownTimer(3600, 50) {
       public void onTick(long millisUntilFinished) {
        cont--;
        angulo = angulo + 0.2;
        Cont2 = Cont2 - 0.4;
        Camaleon.setTranslationX((float) cont);
        Camaleon.setTranslationY((float) Cont2);
        Camaleon.setRotation((float) angulo);
       }
  public void onFinish() {
        cont =Camaleon.getTranslationX();
        Cont2 = Camaleon.getTranslationY();
      new CountDownTimer(12000, 50) {public void onTick(long
millisUntilFinished) {
                    cont--;
                    Cont2 = Cont2 - 0.6;
                    Camaleon.setTranslationX((float) cont);
                    Camaleon.setTranslationY((float) Cont2);
                    }
public void onFinish() {}
}.start();
}
}.start();
}
 }.start();
}
}.start();
}
}.start();


 Camaleon.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
 final PopupMenu popupMenu = new PopupMenu(Portada.this, Camaleon);
  popupMenu.getMenuInflater().inflate(R.menu.menu_popup,
  popupMenu.getMenu());
  popupMenu.setOnMenuItemClickListener(new
  PopupMenu.OnMenuItemClickListener() {
   @Override
    public boolean onMenuItemClick(MenuItem item) {

     switch (item.getItemId()){
      case R.id.Configuracion:
          Intent Cambio=new Intent(Portada.this,DispositivosBT.class);
          startActivity(Cambio);
          break;
      case R.id.Salir:
          finish();
          break;
          }
      return false;
     }
     });
     popupMenu.show();
     }
        });}}
```

DispositivosBT.java

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.content.Intent;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import java.util.Set;

public class DispositivosBT extends AppCompatActivity {

    private static final String TAG = "DispositivosBT";
    ListView IdLista;
    public static String EXTRA_DEVICE_ADDRESS = "device_address";
    private BluetoothAdapter mBtAdapter;
    private ArrayAdapter mPairedDevicesArrayAdapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_dispositivos_bt);

    }
@Override
public void onResume()
{
    super.onResume();
    VerificarEstadoBT();

    mPairedDevicesArrayAdapter = new ArrayAdapter(this,
R.layout.nombre_dispositivos);
    IdLista = (ListView) findViewById(R.id.idLista);
    IdLista.setAdapter(mPairedDevicesArrayAdapter);
    IdLista.setOnItemClickListener(mDeviceClickListener);
    // Obtiene el adaptador local Bluetooth adapter
    mBtAdapter = BluetoothAdapter.getDefaultAdapter();
    Set <BluetoothDevice> pairedDevices =
mBtAdapter.getBondedDevices();
    if (pairedDevices.size() > 0)
    {
        for (BluetoothDevice device : pairedDevices) {

            mPairedDevicesArrayAdapter.add(device.getName() + "\n" +
device.getAddress());
        }
    }
}
```

```
private AdapterView.OnItemClickListener mDeviceClickListener = new
AdapterView.OnItemClickListener() {
        public void onItemClick(AdapterView av, View v, int arg2, long
arg3) {

            String info = ((TextView) v).getText().toString();
            String address = info.substring(info.length() - 17);

            Intent i = new Intent(DispositivosBT.this,
Principal.class);

            i.putExtra(EXTRA_DEVICE_ADDRESS, address);
            startActivity(i);

        }
    };

private void VerificarEstadoBT() {
        // Comprueba que el dispositivo tiene Bluetooth y que está
encendido.
        mBtAdapter= BluetoothAdapter.getDefaultAdapter();
        if(mBtAdapter==null) {
            Toast.makeText(getBaseContext(), "El dispositivo no
soporta Bluetooth", Toast.LENGTH_SHORT).show();
        } else {
            if (mBtAdapter.isEnabled()) {
                Log.d(TAG, "...Bluetooth Activado...");
            } else {
                //Solicita al usuario que active Bluetooth
                Intent enableBtIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
                startActivityForResult(enableBtIntent, 1);

            }
        }
    }
}
```

Portada.java

```
package com.tfg.appcamaleon;

import android.content.Intent;
import android.os.CountDownTimer;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.CheckBox;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.os.Handler;
import android.widget.Toast;
import android.widget.ToggleButton;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.UUID;
import static java.lang.Boolean.FALSE;
import static java.lang.Boolean.TRUE;

public class Principal extends AppCompatActivity {
    TextView
Sens,Otros,Temp,Hum,Hora,Minutos,CalorApagado,PrincipalApagado,Secunda
rioApagado,Encendido;
    ProgressBar Prim,Terc,TemperaturaBar, HumedadBar;
    ImageView Ventilador;
    CheckBox Comp;
    ImageView Conf;
    ImageButton SetHora;
    ToggleButton Luz,Luz2,Calor;
    String a,b,c;
    double MovimientoTemp,MovimientoHum;
    int Horacheck,Mincheck,Calordorm,Encender;
    float Cont;
    //Comunicacion
    public Handler bluetoothIn;
    public final int handlerState = 0;
    private BluetoothAdapter btAdapter = null;
    private BluetoothSocket btSocket = null;
    private StringBuilder DataStringIN = new StringBuilder();
    private ConnectedThread MyConexionBT;
    private static final UUID BTMODULEUUID =
UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");
    private static String address = null;
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_principal);
    Ventilador=(ImageView) findViewById(R.id.idVentilador);
    Sens=(TextView)findViewById(R.id.Sensores);
    Temp=(TextView)findViewById(R.id.Temperatura);
    Hum=(TextView)findViewById(R.id.idHumedad);
    Hora=(TextView)findViewById(R.id.idHora);
    Minutos=(TextView)findViewById(R.id.idMinutos);
    CalorApagado=(TextView)findViewById(R.id.idCalorApagado);
    PrincipalApagado=(TextView)findViewById(R.id.idPrincipalApagado);

SecundarioApagado=(TextView)findViewById(R.id.idSecundariaApagado);
    Encendido=(TextView)findViewById(R.id.idEncender);
    Prim=(ProgressBar)findViewById(R.id.ProgressBar);
    TemperaturaBar=(ProgressBar)findViewById(R.id.idTemperaturaBar);
    HumedadBar=(ProgressBar)findViewById(R.id.idHumedadBar);
    Comp=(CheckBox)findViewById(R.id.idComp);
    Conf=(ImageView)findViewById(R.id.idAjustes);
    Calor=(ToggleButton) findViewById(R.id.idCalor);
    Luz=(ToggleButton)findViewById(R.id.idLuz);
    Luz2=(ToggleButton)findViewById(R.id.idLuz2);
    SetHora=(ImageButton)findViewById(R.id.idHoraEnviar);
    Conf.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent Change = new
Intent(Principal.this,DispositivosBT.class);
            startActivity(Change);
            finish();
        }
    });
    Calor.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {MyConexionBT.write("Q");
        }
    });
    Luz.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {MyConexionBT.write("W");
        }
    });
    Luz2.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) { MyConexionBT.write("R");
        }
    });
    SetHora.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {

if (Hora.getText().length()<2 || Minutos.getText().length() < 2) {
    Toast.makeText(getBaseContext(), "El formato no es correcto",
Toast.LENGTH_LONG).show();
    Hora.setText("");
    Minutos.setText("");
    a = "";
    b = "";
}
```

```
else {
    a = String.valueOf(Hora.getText().subSequence(0, 1)) +
String.valueOf(Hora.getText().subSequence(1, 2));
    b = String.valueOf(Minutos.getText().subSequence(0, 1)) +
String.valueOf(Minutos.getText().subSequence(1, 2));
    c = a + b;
    Horacheck = Integer.parseInt(a);
    Mincheck = Integer.parseInt(b);

if (Horacheck < 24 && Mincheck < 60) {
    MyConexionBT.write(String.valueOf(Hora.getText().subSequence(0,
1)));
    MyConexionBT.write(String.valueOf(Hora.getText().subSequence(1,
2)));
    MyConexionBT.write(String.valueOf(Minutos.getText().subSequence(0,
1)));
    MyConexionBT.write(String.valueOf(Minutos.getText().subSequence(1,
2)));

} else {
    Toast.makeText(getBaseContext(), "Los minutos o las horas no son
correctos", Toast.LENGTH_LONG).show();
    Hora.setText("");
    Minutos.setText("");
}

PrincipalApagado.setText(a + ":" + b);
SecundarioApagado.setText(a + ":" + b);
Calordorm = Integer.parseInt(c) + 200;


if (Calordorm > 2390) {
    Calordorm = Calordorm - 2400;
    switch (String.valueOf(Calordorm).length()) {
        case 0:
            CalorApagado.setText("00:00");
            break;
        case 1:
            CalorApagado.setText("00:0" + String.valueOf(Calordorm));
            break;
        case 2:
            CalorApagado.setText("00:" + String.valueOf(Calordorm));
            break;
        case 3:
            CalorApagado.setText("0" +
String.valueOf(Calordorm).substring(0, 1) + ":" +
String.valueOf(Calordorm).substring(1, 3));
            break;
        case 4:

CalorApagado.setText(String.valueOf(Calordorm).substring(0, 2) + ":" +
String.valueOf(Calordorm).substring(2, 4));
            break;
    }
}
```

```
else {
    switch (String.valueOf(Calordorm).length()) {
        case 0:
            CalorApagado.setText("00:00");
            break;
        case 1:
            CalorApagado.setText("00:0" + String.valueOf(Calordorm));
            break;
        case 2:
            CalorApagado.setText("00:" + String.valueOf(Calordorm));
            break;
        case 3:
            CalorApagado.setText("0" +
String.valueOf(Calordorm).substring(0, 1) + ":" +
String.valueOf(Calordorm).substring(1, 3));
            break;
        case 4:

CalorApagado.setText(String.valueOf(Calordorm).substring(0, 2) + ":" +
String.valueOf(Calordorm).substring(2, 4));
            break;
    }
}
Encender = Integer.parseInt(c) + 1100;
if (Encender > 2390) {
    Encender = Encender - 2400;
    switch (String.valueOf(Encender).length()) {
        case 0:
            Encendido.setText("00:00");
            break;
        case 1:
            Encendido.setText("00:0" + String.valueOf(Encender));
            break;
        case 2:
            Encendido.setText("00:" + String.valueOf(Encender));
            break;
        case 3:
            Encendido.setText("0" +
String.valueOf(Encender).substring(0, 1) + ":" +
String.valueOf(Encender).substring(1, 3));
            break;
        case 4:
            Encendido.setText(String.valueOf(Encender).substring(0, 2)
+ ":" + String.valueOf(Encender).substring(2, 4));
            break;
    }
}
```

```
else {
                    switch (String.valueOf(Encender).length()) {
                        case 0:
                            Encendido.setText("00:00");
                            break;
                        case 1:
                            Encendido.setText("00:0" +
String.valueOf(Encender));
                            break;
                        case 2:
                            Encendido.setText("00:" +
String.valueOf(Encender));
                            break;
                        case 3:
                            Encendido.setText("0" +
String.valueOf(Encender).substring(0, 1) + ":" +
String.valueOf(Encender).substring(1, 3));
                            break;
                        case 4:

Encendido.setText(String.valueOf(Encender).substring(0, 2) + ":" +
String.valueOf(Encender).substring(2, 4));
                            break;
                    }
                }

            }
        }
    });
    Ventilador.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {MyConexionBT.write("V");

        }
    });

}


@Override
public void onStart(){
    super.onStart();

    bluetoothIn = new Handler() {
        public void handleMessage(android.os.Message msg) {
            if (msg.what == handlerState) {
                String readMessage = (String) msg.obj;
                DataStringIN.append(readMessage);

                String
Select=DataStringIN.substring(DataStringIN.length()-
1,DataStringIN.length());
```

```
switch (Select){
    case "#":
        if (DataStringIN.length()==9) {

            String dataInPrint =
DataStringIN.substring(DataStringIN.length() - 4,
DataStringIN.length() - 2);//Temperatura
            String dataInPrint3 =
DataStringIN.substring(DataStringIN.length() - 6,
DataStringIN.length() - 4);//Humedad
            String dataInPrint4 = DataStringIN.substring(0,
DataStringIN.length() - 6); //Estado de luzes
            String dataInPrint2 =
DataStringIN.substring(DataStringIN.length() - 2,
DataStringIN.length() - 1);//Ventilado
            Temp.setText("<-" + dataInPrint + "°C");
            MovimientoTemp = (7.575 *
Double.parseDouble(String.valueOf(dataInPrint)));

            Hum.setText("<-" + dataInPrint3 + "%");
            MovimientoHum = (3.03 *
Double.parseDouble(String.valueOf(dataInPrint3)));


TemperaturaBar.setProgress(Integer.parseInt(String.valueOf(dataInPrint
)));
HumedadBar.setProgress(Integer.parseInt(dataInPrint3));
switch (dataInPrint2){
    case "1":
        new CountDownTimer(5000, 50) {
        public void onTick(long millisUntilFinished) {
            Cont = Cont + 2;
            Ventilador.setRotation(Cont);
        }

        public void onFinish() {

        }
         }.start();
        break;
}
switch (dataInPrint4) {
        case "000":
            Calor.setChecked(FALSE);
            Luz.setChecked(FALSE);
            Luz2.setChecked(FALSE);
            break;
        case "001":
            Calor.setChecked(TRUE);
            Luz.setChecked(FALSE);
            Luz2.setChecked(FALSE);
            break;
        case "002":
            Calor.setChecked(FALSE);
            Luz.setChecked(TRUE);
            Luz2.setChecked(FALSE);
            break;
        case "003":
            Calor.setChecked(TRUE);
            Luz.setChecked(TRUE);
```

```
                Luz2.setChecked(FALSE);
                break;
            case "004":
                Calor.setChecked(FALSE);
                Luz.setChecked(FALSE);
                Luz2.setChecked(TRUE);
                break;
            case "005":
                Calor.setChecked(TRUE);
                Luz.setChecked(FALSE);
                Luz2.setChecked(TRUE);
                break;
            case "006":
                Calor.setChecked(FALSE);
                Luz.setChecked(TRUE);
                Luz2.setChecked(TRUE);
                break;
            case "007":
                Calor.setChecked(TRUE);
                Luz.setChecked(TRUE);
                Luz2.setChecked(TRUE);
                break;

        }
    }
    Temp.setTranslationY(-1*(int)MovimientoTemp);
    Hum.setTranslationY(-1*(int)MovimientoHum);

    DataStringIN.delete(0, DataStringIN.length());
    try {
        MyConexionBT.write("T");
    }catch (Exception e){
        Toast.makeText(getBaseContext(), "La Conexión fallo",
    Toast.LENGTH_LONG).show();
    }
    break;
    case "$":
        String dataInPrint5 = DataStringIN.substring(0,
    DataStringIN.length() - 1);

        Comp.setChecked(TRUE);
        Comp.setText("Conectado");

    PrincipalApagado.setText(dataInPrint5.substring(0,2)+":"+dataInPrint5.
    substring(2,4));

    SecundarioApagado.setText(dataInPrint5.substring(0,2)+":"+dataInPrint5
    .substring(2,4));

    Calordorm=Integer.parseInt(dataInPrint5)+200;
```

```
if(Calordorm>2390){
    Calordorm=Calordorm-2400;
    switch (String.valueOf(Calordorm).length()){
        case 0:
            CalorApagado.setText("00:00");
            break;
        case 1:
            CalorApagado.setText("00:0"+String.valueOf(Calordorm));
            break;
        case 2:
            CalorApagado.setText("00:"+String.valueOf(Calordorm));
            break;
        case 3:

CalorApagado.setText("0"+String.valueOf(Calordorm).substring(0,1)+":"+
String.valueOf(Calordorm).substring(1,3));
            break;
        case 4:

CalorApagado.setText(String.valueOf(Calordorm).substring(0,2)+":"+Stri
ng.valueOf(Calordorm).substring(2,4));
            break;
    }
}else
{switch (String.valueOf(Calordorm).length()){
    case 0:
        CalorApagado.setText("00:00");
        break;
    case 1:
        CalorApagado.setText("00:0"+String.valueOf(Calordorm));
        break;
    case 2:
        CalorApagado.setText("00:"+String.valueOf(Calordorm));
        break;
    case 3:

CalorApagado.setText("0"+String.valueOf(Calordorm).substring(0,1)+":"+
String.valueOf(Calordorm).substring(1,3));
        break;
    case 4:

CalorApagado.setText(String.valueOf(Calordorm).substring(0,2)+":"+Stri
ng.valueOf(Calordorm).substring(2,4));
        break;
}
}

Encender=Integer.parseInt(dataInPrint5)+1100;
if(Encender>2390){
    Encender=Encender-2400;
    switch (String.valueOf(Encender).length()){
        case 0:
            Encendido.setText("00:00");
            break;
        case 1:
            Encendido.setText("00:0"+String.valueOf(Encender));
            break;
        case 2:
            Encendido.setText("00:"+String.valueOf(Encender));
            break;
```

```
            case 3:

Encendido.setText("0"+String.valueOf(Encender).substring(0,1)+":"+Stri
ng.valueOf(Encender).substring(1,3));
                break;
            case 4:

Encendido.setText(String.valueOf(Encender).substring(0,2)+":"+String.v
alueOf(Encender).substring(2,4));
                break;
        }
    }else {
        switch (String.valueOf(Encender).length()){
            case 0:
                Encendido.setText("00:00");
                break;
            case 1:
                Encendido.setText("00:0"+String.valueOf(Encender));
                break;
            case 2:
                Encendido.setText("00:"+String.valueOf(Encender));
                break;
            case 3:

Encendido.setText("0"+String.valueOf(Encender).substring(0,1)+":"+Stri
ng.valueOf(Encender).substring(1,3));
                break;
            case 4:

Encendido.setText(String.valueOf(Encender).substring(0,2)+":"+String.v
alueOf(Encender).substring(2,4));
                break;
        }
    }
    DataStringIN.delete(0, DataStringIN.length());
    try {
        MyConexionBT.write("T");
    }catch (Exception e){
        Toast.makeText(getBaseContext(), "La Conexión fallo",
    Toast.LENGTH_LONG).show();
    }
    }
    }
    }
    };
       btAdapter = BluetoothAdapter.getDefaultAdapter();
        VerificarEstadoBT();

    }
```

```
public BluetoothSocket createBluetoothSocket(BluetoothDevice device)
throws IOException
{
    return device.createRfcommSocketToServiceRecord(BTMODULEUUID);
}

@Override
public void onResume()
{
    super.onResume();
    Intent intent = getIntent();
    address =
intent.getStringExtra(DispositivosBT.EXTRA_DEVICE_ADDRESS);
    BluetoothDevice device = btAdapter.getRemoteDevice(address);

    try
    {
        btSocket = createBluetoothSocket(device);
    } catch (IOException e) {
        Toast.makeText(getBaseContext(), "La creacción del Socket
fallo", Toast.LENGTH_LONG).show();
    }
    // Establece la conexión con el socket Bluetooth.
try
    {
        btSocket.connect();
    } catch (IOException e) {
        try {
            btSocket.close();
        } catch (IOException e2) {}
    }
    MyConexionBT = new ConnectedThread(btSocket);
    MyConexionBT.start();
    try {
        MyConexionBT.write("c");

    }catch (Exception e){
        Toast.makeText(getBaseContext(), "La Conexión fallo",
Toast.LENGTH_LONG).show();
    }
}

@Override
public void onPause()
{
    super.onPause();
   try{ // Cuando se sale de la aplicación esta parte permite
        // que no se deje abierto el socket
       btSocket.close();
       finish();
   } catch (IOException e2) {}
}
```

```
//Comprueba que el dispositivo Bluetooth Bluetooth está disponible y
solicita que se active si está desactivado
public void VerificarEstadoBT() {

    if(btAdapter==null) {
        Toast.makeText(getBaseContext(), "El dispositivo no soporta
bluetooth", Toast.LENGTH_LONG).show();
    } else {
        if (btAdapter.isEnabled()) {
        } else {
            Intent enableBtIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
            startActivityForResult(enableBtIntent, 1);
        }
    }
}


private class ConnectedThread extends Thread
{
private final InputStream mmInStream;
private final OutputStream mmOutStream;

public ConnectedThread(BluetoothSocket socket)
{
    InputStream tmpIn = null;
    OutputStream tmpOut = null;
    try
    {
        tmpIn = socket.getInputStream();
        tmpOut = socket.getOutputStream();
    } catch (IOException e) { }
    mmInStream = tmpIn;
    mmOutStream = tmpOut;
}
public void run()
{
byte[] buffer = new byte[256];
int bytes;

// Se mantiene en modo escucha para determinar el ingreso de datos
while (true) {
try {
    bytes = mmInStream.read(buffer);
    String readMessage = new String(buffer, 0, bytes);
    // Envia los datos obtenidos hacia el evento via handler
    bluetoothIn.obtainMessage(handlerState, bytes, -1,
readMessage).sendToTarget();

} catch (IOException e) {
    Toast.makeText(getBaseContext(), "La Conexión fallo",
Toast.LENGTH_LONG).show();
    break;
}
}
}
```

```
public void write(String input)
{
    try {
        mmOutStream.write(input.getBytes());
    }
    catch (IOException e)
    {
        Toast.makeText(getBaseContext(), "La Conexión fallo",
Toast.LENGTH_LONG).show();
        finish();
    }
}
}
}
```

FondoControl.xml

```
<?xml version="1.0" encoding="utf-8"?>
<layer-list
xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@android:id/background">
        <shape>
            <corners android:radius="5dip" />
            <gradient
                android:startColor="@color/md_green_A700"
                android:centerColor="@color/md_brown_300"
                android:endColor="@color/md_brown_500"
                android:angle="270"
                />
        </shape>
    </item>
</layer-list>
```

Menu_popup.xml

```
?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <item
        android:id="@+id/Configuracion"
        android:title="Iniciar" />
    <item android:id="@+id/Salir"
        android:title="Salir"/>
</menu>
```

Humedad.xml

```
<?xml version="1.0" encoding="utf-8"?>
<layer-list
xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@android:id/background">
        <shape>
            <corners android:radius="5dip" />
            <gradient
                android:background="@android:color/transparent"
                android:angle="270"
                />
        </shape>
    </item>

    <item android:id="@android:id/secondaryProgress">
        <clip>
            <shape>
                <corners android:radius="5dip" />
                <gradient
                    android:startColor="#80ffd300"
                    android:centerColor="#80ffb600"
                    android:centerY="0.75"
                    android:endColor="#a0ffcb00"
                    android:angle="270"
                    />
            </shape>
        </clip>
    </item>
    <item
        android:id="@android:id/progress"
        >

        <clip>
            <shape>
                <corners
                    android:radius="6dip" />
                <gradient
                    android:startColor="@color/md_white_1000"
                    android:centerColor="@color/md_blue_200"
                    android:endColor="@color/md_blue_A700"
                    />
            </shape>
        </clip>
    </item>
</layer-list>
```

Temperatura.xml

```
<?xml version="1.0" encoding="utf-8"?>
<layer-list
xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@android:id/background">
        <shape>
            <corners android:radius="5dip" />
            <gradient
                android:background="@android:color/transparent"
                android:angle="270"
                />
        </shape>
    </item>

    <item android:id="@android:id/secondaryProgress">
        <clip>
            <shape>
                <corners android:radius="5dip" />
                <gradient
                    android:startColor="#80ffd300"
                    android:centerColor="#80ffb600"
                    android:centerY="0.75"
                    android:endColor="#a0ffcb00"
                    android:angle="270"
                    />
            </shape>
        </clip>
    </item>
    <item
        android:id="@android:id/progress"
        >

        <clip>
            <shape>
                <corners
                    android:radius="6dip" />
                <gradient
                    android:startColor="@color/md_blue_A700"
                    android:centerColor="@color/md_deep_orange_600"
                    android:endColor="@color/md_red_900"
                     />
            </shape>
        </clip>
    </item>
</layer-list>
```

activity_portada.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    android:background="@drawable/bosque"
    tools:context=".Portada">

    <TextView
        android:id="@+id/textView"
        style="@android:style/Animation"
        android:layout_width="268dp"
        android:layout_height="61dp"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_marginStart="73dp"
        android:layout_marginTop="82dp"
        android:shadowColor="@color/md_amber_200"
        android:text="App Camaleón"
        android:textColor="@color/md_black_1000"
        android:textSize="36sp"
        android:textStyle="bold|italic" />

    <ImageView
        android:id="@+id/imageView4"
        android:layout_width="403dp"
        android:layout_height="306dp"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_marginStart="0dp"
        android:layout_marginTop="204dp"
        android:rotation="50"
        app:srcCompat="@drawable/palo" />

    <ImageView
        android:id="@+id/idFotoCamaleon"
        android:layout_width="116dp"
        android:layout_height="83dp"
        android:layout_marginLeft="295dp"
        android:layout_marginTop="360dp"
        android:contentDescription="TODO"
        android:rotation="40"
        android:visibility="visible"
        app:srcCompat="@drawable/cam" />

</RelativeLayout>
```

activity_dispositivos_bt.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/idBlue"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".DispositivosBT">

    <TextView
        android:id="@+id/idTitulo"
        android:layout_width="254dp"
        android:layout_height="66dp"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_marginStart="90dp"
        android:layout_marginTop="11dp"
        android:gravity="center"
        android:text="Configuración"
        android:textSize="24sp" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="match_parent"
        android:layout_height="40dp"
        android:gravity="fill_vertical"
        android:text="    Dispositivos Bluetooth vinculados"
        android:textSize="17dp" />

    <ListView
        android:id="@+id/idLista"

style="@android:style/Widget.DeviceDefault.Light.ListView.DropDown"
        android:layout_width="match_parent"
        android:layout_height="250dp"
        android:layout_below="@+id/idTitulo"
        android:layout_alignParentBottom="true"
        android:layout_marginBottom="426dp" />

</LinearLayout>
```

activity_principal.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:backgroundTint="@android:color/transparent"
    android:foregroundTint="@android:color/transparent"
    tools:context=".Principal">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="528dp"
        android:layout_marginTop="50dp"
        android:background="@drawable/fondocontrol">

        <ToggleButton
            android:id="@+id/idCalor"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="30dp"
            android:layout_marginTop="355dp"
            android:checked="true"
            android:text="@string/togglebutton"
            android:textOff="@string/apagado"
            android:textOn="Encendido" />

        <ToggleButton
            android:id="@+id/idLuz"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="166dp"
            android:layout_marginTop="355dp"
            android:checked="true"
            android:text="ToggleButton"
            android:textOff="Apagado"
            android:textOn="Encendido" />

        <ToggleButton
            android:id="@+id/idLuz2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="291dp"
            android:layout_marginTop="355dp"
            android:checked="true"
            android:text="ToggleButton"
            android:textOff="Apagado"
            android:textOn="Encendido" />
```

```
        <TextView
            android:id="@+id/idTemperatura"
            android:layout_width="100dp"
            android:layout_height="35dp"
            android:layout_marginStart="11dp"
            android:layout_marginTop="15dp"
            android:text="@string/temperatura"

android:textAppearance="@style/TextAppearance.AppCompat.Body2"
            android:visibility="visible" />

        <TextView
            android:id="@+id/idHumedadText"
            android:layout_width="100dp"
            android:layout_height="35dp"
            android:layout_marginStart="199dp"
            android:layout_marginTop="15dp"
            android:text="Humedad:"

android:textAppearance="@style/TextAppearance.AppCompat.Body2"
            android:visibility="visible" />

        <ProgressBar
            android:id="@+id/idTemperaturaBar"
            style="@android:style/Widget.ProgressBar.Horizontal"
            android:layout_width="116dp"
            android:layout_height="32dp"
            android:layout_marginStart="1dp"
            android:layout_marginLeft="300dp"
            android:layout_marginTop="109dp"
            android:max="40"
            android:progress="40"
            android:progressDrawable="@drawable/temperatura"
            android:rotation="-90"
            android:visibility="visible" />

        <ProgressBar
            android:id="@+id/idHumedadBar"
            style="@android:style/Widget.ProgressBar.Horizontal"
            android:layout_width="116dp"
            android:layout_height="32dp"
            android:layout_marginStart="200dp"
            android:layout_marginLeft="300dp"
            android:layout_marginTop="109dp"
            android:max="100"
            android:progress="50"
            android:progressDrawable="@drawable/humedad"
            android:rotation="-90"
            android:visibility="visible" />

        <TextView
            android:id="@+id/idHumedad"
            android:layout_width="61dp"
            android:layout_height="33dp"
            android:layout_marginStart="280dp"
            android:layout_marginTop="172dp"
            android:editable="false"

android:textAppearance="@style/TextAppearance.AppCompat.Body2" />

        <TextView
```

```
            android:id="@+id/Temperatura"
            android:layout_width="59dp"
            android:layout_height="32dp"
            android:layout_marginStart="83dp"
            android:layout_marginTop="172dp"

android:textAppearance="@style/TextAppearance.AppCompat.Body2" />

        <TextView
            android:id="@+id/idHoraString"
            android:layout_width="125dp"
            android:layout_height="30dp"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="32dp"
            android:layout_marginTop="216dp"
            android:text="Hora de Dormir:"
            android:textColor="@color/md_white_1000"
            android:visibility="visible" />

        <TextView
            android:id="@+id/textView3"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="63dp"
            android:layout_marginTop="335dp"
            android:text="Calor"

android:textAppearance="@style/TextAppearance.AppCompat.Body2"
            android:textSize="16sp"
            android:visibility="visible" />

        <TextView
            android:id="@+id/textView8"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="109dp"
            android:layout_marginTop="410dp"
            android:text="Calor"

android:textAppearance="@style/TextAppearance.AppCompat.Body2"
            android:visibility="visible" />

        <TextView
            android:id="@+id/textView4"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="177dp"
            android:layout_marginTop="331dp"
            android:text="Luz Principal"

android:textAppearance="@style/TextAppearance.AppCompat.Body2"
            android:textSize="16sp" />

        <TextView
```

```
            android:id="@+id/textView9"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="198dp"
            android:layout_marginTop="410dp"
            android:text="Luz Principal"

android:textAppearance="@style/TextAppearance.AppCompat.Body2" />

        <TextView
            android:id="@+id/textView5"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="293dp"
            android:layout_marginTop="330dp"
            android:text="Luz Secundaria"

android:textAppearance="@style/TextAppearance.AppCompat.Body2"
            android:textSize="16sp" />

        <TextView
            android:id="@+id/textView10"
            android:layout_width="101dp"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="310dp"
            android:layout_marginTop="410dp"
            android:text="Luz Secundaria"

android:textAppearance="@style/TextAppearance.AppCompat.Body2" />

        <EditText
            android:id="@+id/idHora"
            android:layout_width="42dp"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="30dp"
            android:layout_marginTop="260dp"
            android:ems="10"
            android:inputType="number" />

        <EditText
            android:id="@+id/idMinutos"
            android:layout_width="42dp"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="79dp"
            android:layout_marginTop="260dp"
            android:ems="10"
            android:inputType="number" />


        <ImageButton
            android:id="@+id/idHoraEnviar"
```

```
            android:layout_width="86dp"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="135dp"
            android:layout_marginTop="259dp"
            app:srcCompat="@android:drawable/ic_menu_send" />

        <ImageView
            android:id="@+id/idVentilador"
            android:layout_width="182dp"
            android:layout_height="108dp"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="229dp"
            android:layout_marginTop="217dp"
            app:srcCompat="@android:drawable/ic_menu_compass" />

        <TextView
            android:id="@+id/textView6"
            android:layout_width="63dp"
            android:layout_height="38dp"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="9dp"
            android:layout_marginTop="430dp"
            android:text="Hora de apagado:"

android:textAppearance="@style/TextAppearance.AppCompat.Body2" />

        <TextView
            android:id="@+id/textView7"
            android:layout_width="82dp"
            android:layout_height="37dp"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="9dp"
            android:layout_marginTop="479dp"
            android:text="Hora de encendido:"

android:textAppearance="@style/TextAppearance.AppCompat.Body2" />

        <TextView
            android:id="@+id/idCalorApagado"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="109dp"
            android:layout_marginTop="440dp" />

        <TextView
            android:id="@+id/idPrincipalApagado"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="215dp"
            android:layout_marginTop="440dp" />

        <TextView
```

```
            android:id="@+id/idSecundariaApagado"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="330dp"
            android:layout_marginTop="440dp" />

        <TextView
            android:id="@+id/idEncender"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="215dp"
            android:layout_marginTop="493dp" />

    </RelativeLayout>

    <ProgressBar
        android:id="@+id/ProgressBar"
        style="?android:attr/progressBarStyleHorizontal"
        android:layout_width="245dp"
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_marginStart="91dp"
        android:layout_marginTop="32dp"
        android:max="1"
        android:progress="1"
        android:progressTint="@color/md_light_green_A400"
        android:visibility="visible" />

    <android.support.v7.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_width="match_parent"
        android:layout_height="80dp"
        android:layout_alignParentTop="true"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginTop="579dp"
        android:layout_marginEnd="0dp"
        android:layout_marginBottom="0dp"
        android:background="?attr/colorPrimary"
        android:minHeight="250dp"
        android:theme="?attr/actionBarTheme" />

    <CheckBox
        android:id="@+id/idComp"
        android:layout_width="161dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="250dp"
        android:layout_marginTop="599dp"
        android:text="Desconectado" />



    <TextView
        android:id="@+id/Sensores"
        android:layout_width="244dp"
```

```
            android:layout_height="wrap_content"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="93dp"
            android:layout_marginTop="0dp"
            android:fontFamily="sans-serif"
            android:text="Panel de Control"
            android:textAlignment="center"
            android:textColor="@color/md_light_green_A400"
            android:textSize="30sp" />

        <ImageView
            android:id="@+id/idAjustes"
            android:layout_width="46dp"
            android:layout_height="39dp"
            android:layout_alignParentStart="true"
            android:layout_alignParentTop="true"
            android:layout_marginStart="24dp"
            android:layout_marginTop="597dp"
            app:srcCompat="@drawable/ajustes" />

</RelativeLayout>
```

colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <array name="reds">
        <item>@color/md_red_500</item>
        <item>@color/md_red_50</item>
        <item>@color/md_red_100</item>
        <item>@color/md_red_200</item>
        <item>@color/md_red_300</item>
        <item>@color/md_red_400</item>
        <item>@color/md_red_600</item>
        <item>@color/md_red_700</item>
        <item>@color/md_red_800</item>
        <item>@color/md_red_900</item>
        <item>@color/md_red_A100</item>
        <item>@color/md_red_A200</item>
        <item>@color/md_red_A400</item>
        <item>@color/md_red_A700</item>
    </array>
    <color name="md_red_50">#ffebee</color>
    <color name="md_red_100">#ffcdd2</color>
    <color name="md_red_200">#ef9a9a</color>
    <color name="md_red_300">#e57373</color>
    <color name="md_red_400">#ef5350</color>
    <color name="md_red_500">#f44336</color>
    <color name="md_red_600">#e53935</color>
    <color name="md_red_700">#d32f2f</color>
    <color name="md_red_800">#c62828</color>
    <color name="md_red_900">#b71c1c</color>
    <color name="md_red_A100">#ff8a80</color>
    <color name="md_red_A200">#ff5252</color>
    <color name="md_red_A400">#ff1744</color>
    <color name="md_red_A700">#d50000</color>
    <color name="md_red_500_25">#40f44336</color>
    <color name="md_red_500_50">#80f44336</color>
    <color name="md_red_500_75">#C0F44336</color>
    <!-- pinks -->
    <array name="pinks">
        <item>@color/md_pink_500</item>
        <item>@color/md_pink_50</item>
        <item>@color/md_pink_100</item>
        <item>@color/md_pink_200</item>
        <item>@color/md_pink_300</item>
        <item>@color/md_pink_400</item>
        <item>@color/md_pink_600</item>
        <item>@color/md_pink_700</item>
        <item>@color/md_pink_800</item>
        <item>@color/md_pink_900</item>
        <item>@color/md_pink_A100</item>
        <item>@color/md_pink_A200</item>
        <item>@color/md_pink_A400</item>
        <item>@color/md_pink_A700</item>
    </array>
    <color name="md_pink_50">#fce4ec</color>
    <color name="md_pink_100">#f8bbd0</color>
    <color name="md_pink_200">#f48fb1</color>
    <color name="md_pink_300">#f06292</color>
    <color name="md_pink_400">#ec407a</color>
    <color name="md_pink_500">#e91e63</color>
    <color name="md_pink_600">#d81b60</color>
```

```
<color name="md_pink_700">#c2185b</color>
<color name="md_pink_800">#ad1457</color>
<color name="md_pink_900">#880e4f</color>
<color name="md_pink_A100">#ff80ab</color>
<color name="md_pink_A200">#ff4081</color>
<color name="md_pink_A400">#f50057</color>
<color name="md_pink_A700">#c51162</color>
<color name="md_pink_500_25">#40e91e63</color>
<color name="md_pink_500_50">#80e91e63</color>
<color name="md_pink_500_75">#c0e91e63</color>
<!-- purples -->
<array name="purples">
    <item>@color/md_purple_500</item>
    <item>@color/md_purple_50</item>
    <item>@color/md_purple_100</item>
    <item>@color/md_purple_200</item>
    <item>@color/md_purple_300</item>
    <item>@color/md_purple_400</item>
    <item>@color/md_purple_600</item>
    <item>@color/md_purple_700</item>
    <item>@color/md_purple_800</item>
    <item>@color/md_purple_900</item>
    <item>@color/md_purple_A100</item>
    <item>@color/md_purple_A200</item>
    <item>@color/md_purple_A400</item>
    <item>@color/md_purple_A700</item>
</array>
<color name="md_purple_50">#f3e5f5</color>
<color name="md_purple_100">#e1bee7</color>
<color name="md_purple_200">#ce93d8</color>
<color name="md_purple_300">#ba68c8</color>
<color name="md_purple_400">#ab47bc</color>
<color name="md_purple_500">#9c27b0</color>
<color name="md_purple_600">#8e24aa</color>
<color name="md_purple_700">#7b1fa2</color>
<color name="md_purple_800">#6a1b9a</color>
<color name="md_purple_900">#4a148c</color>
<color name="md_purple_A100">#ea80fc</color>
<color name="md_purple_A200">#e040fb</color>
<color name="md_purple_A400">#d500f9</color>
<color name="md_purple_A700">#aa00ff</color>
<color name="md_purple_500_25">#409c27b0</color>
<color name="md_purple_500_50">#809c27b0</color>
<color name="md_purple_500_75">#c09c27b0</color>
<!-- deep purples -->
<array name="deep_purples">
    <item>@color/md_deep_purple_500</item>
    <item>@color/md_deep_purple_50</item>
    <item>@color/md_deep_purple_100</item>
    <item>@color/md_deep_purple_200</item>
    <item>@color/md_deep_purple_300</item>
    <item>@color/md_deep_purple_400</item>
    <item>@color/md_deep_purple_600</item>
    <item>@color/md_deep_purple_700</item>
    <item>@color/md_deep_purple_800</item>
    <item>@color/md_deep_purple_900</item>
    <item>@color/md_deep_purple_A100</item>
    <item>@color/md_deep_purple_A200</item>
    <item>@color/md_deep_purple_A400</item>
    <item>@color/md_deep_purple_A700</item>
</array>
```

```
<color name="md_deep_purple_50">#ede7f6</color>
<color name="md_deep_purple_100">#d1c4e9</color>
<color name="md_deep_purple_200">#b39ddb</color>
<color name="md_deep_purple_300">#9575cd</color>
<color name="md_deep_purple_400">#7e57c2</color>
<color name="md_deep_purple_500">#673ab7</color>
<color name="md_deep_purple_600">#5e35b1</color>
<color name="md_deep_purple_700">#512da8</color>
<color name="md_deep_purple_800">#4527a0</color>
<color name="md_deep_purple_900">#311b92</color>
<color name="md_deep_purple_A100">#b388ff</color>
<color name="md_deep_purple_A200">#7c4dff</color>
<color name="md_deep_purple_A400">#651fff</color>
<color name="md_deep_purple_A700">#6200ea</color>
<color name="md_deep_purple_500_25">#40673ab7</color>
<color name="md_deep_purple_500_50">#80673ab7</color>
<color name="md_deep_purple_500_75">#c0673ab7</color>
<!-- indigos -->
<array name="indigos">
    <item>@color/md_indigo_500</item>
    <item>@color/md_indigo_50</item>
    <item>@color/md_indigo_100</item>
    <item>@color/md_indigo_200</item>
    <item>@color/md_indigo_300</item>
    <item>@color/md_indigo_400</item>
    <item>@color/md_indigo_600</item>
    <item>@color/md_indigo_700</item>
    <item>@color/md_indigo_800</item>
    <item>@color/md_indigo_900</item>
    <item>@color/md_indigo_A100</item>
    <item>@color/md_indigo_A200</item>
    <item>@color/md_indigo_A400</item>
    <item>@color/md_indigo_A700</item>
</array>
<color name="md_indigo_50">#e8eaf6</color>
<color name="md_indigo_100">#c5cae9</color>
<color name="md_indigo_200">#9fa8da</color>
<color name="md_indigo_300">#7986cb</color>
<color name="md_indigo_400">#5c6bc0</color>
<color name="md_indigo_500">#3f51b5</color>
<color name="md_indigo_600">#3949ab</color>
<color name="md_indigo_700">#303f9f</color>
<color name="md_indigo_800">#283593</color>
<color name="md_indigo_900">#1a237e</color>
<color name="md_indigo_A100">#8c9eff</color>
<color name="md_indigo_A200">#536dfe</color>
<color name="md_indigo_A400">#3d5afe</color>
<color name="md_indigo_A700">#304ffe</color>
<color name="md_indigo_500_25">#403f51b5</color>
<color name="md_indigo_500_50">#803f51b5</color>
<color name="md_indigo_500_75">#c03f51b5</color>
<!--blues-->
<array name="blues">
    <item>@color/md_blue_500</item>
    <item>@color/md_blue_50</item>
    <item>@color/md_blue_100</item>
    <item>@color/md_blue_200</item>
    <item>@color/md_blue_300</item>
    <item>@color/md_blue_400</item>
    <item>@color/md_blue_600</item>
    <item>@color/md_blue_700</item>
```

```
        <item>@color/md_blue_800</item>
        <item>@color/md_blue_900</item>
        <item>@color/md_blue_A100</item>
        <item>@color/md_blue_A200</item>
        <item>@color/md_blue_A400</item>
        <item>@color/md_blue_A700</item>
    </array>
    <color name="md_blue_50">#e3f2fd</color>
    <color name="md_blue_100">#bbdefb</color>
    <color name="md_blue_200">#90caf9</color>
    <color name="md_blue_300">#64b5f6</color>
    <color name="md_blue_400">#42a5f5</color>
    <color name="md_blue_500">#2196f3</color>
    <color name="md_blue_600">#1e88e5</color>
    <color name="md_blue_700">#1976d2</color>
    <color name="md_blue_800">#1565c0</color>
    <color name="md_blue_900">#0d47a1</color>
    <color name="md_blue_A100">#82b1ff</color>
    <color name="md_blue_A200">#448aff</color>
    <color name="md_blue_A400">#2979ff</color>
    <color name="md_blue_A700">#2962ff</color>
    <color name="md_blue_500_25">#402196f3</color>
    <color name="md_blue_500_50">#802196f3</color>
    <color name="md_blue_500_75">#c02196f3</color>
    <!-- light blues-->
    <array name="light_blues">
        <item>@color/md_light_blue_500</item>
        <item>@color/md_light_blue_50</item>
        <item>@color/md_light_blue_100</item>
        <item>@color/md_light_blue_200</item>
        <item>@color/md_light_blue_300</item>
        <item>@color/md_light_blue_400</item>
        <item>@color/md_light_blue_600</item>
        <item>@color/md_light_blue_700</item>
        <item>@color/md_light_blue_800</item>
        <item>@color/md_light_blue_900</item>
        <item>@color/md_light_blue_A100</item>
        <item>@color/md_light_blue_A200</item>
        <item>@color/md_light_blue_A400</item>
        <item>@color/md_light_blue_A700</item>
    </array>
    <color name="md_light_blue_50">#e1f5fe</color>
    <color name="md_light_blue_100">#b3e5fc</color>
    <color name="md_light_blue_200">#81d4fa</color>
    <color name="md_light_blue_300">#4fc3f7</color>
    <color name="md_light_blue_400">#29b6f6</color>
    <color name="md_light_blue_500">#03a9f4</color>
    <color name="md_light_blue_600">#039be5</color>
    <color name="md_light_blue_700">#0288d1</color>
    <color name="md_light_blue_800">#0277bd</color>
    <color name="md_light_blue_900">#01579b</color>
    <color name="md_light_blue_A100">#80d8ff</color>
    <color name="md_light_blue_A200">#40c4ff</color>
    <color name="md_light_blue_A400">#00b0ff</color>
    <color name="md_light_blue_A700">#0091ea</color>
    <color name="md_light_blue_500_25">#4003a9f4</color>
    <color name="md_light_blue_500_50">#8003a9f4</color>
    <color name="md_light_blue_500_75">#c003a9f4</color>
    <!-- cyans -->
    <array name="cyans">
        <item>@color/md_cyan_500</item>
```

```
        <item>@color/md_cyan_50</item>
        <item>@color/md_cyan_100</item>
        <item>@color/md_cyan_200</item>
        <item>@color/md_cyan_300</item>
        <item>@color/md_cyan_400</item>
        <item>@color/md_cyan_600</item>
        <item>@color/md_cyan_700</item>
        <item>@color/md_cyan_800</item>
        <item>@color/md_cyan_900</item>
        <item>@color/md_cyan_A100</item>
        <item>@color/md_cyan_A200</item>
        <item>@color/md_cyan_A400</item>
        <item>@color/md_cyan_A700</item>
    </array>
    <color name="md_cyan_50">#e0f7fa</color>
    <color name="md_cyan_100">#b2ebf2</color>
    <color name="md_cyan_200">#80deea</color>
    <color name="md_cyan_300">#4dd0e1</color>
    <color name="md_cyan_400">#26c6da</color>
    <color name="md_cyan_500">#00bcd4</color>
    <color name="md_cyan_600">#00acc1</color>
    <color name="md_cyan_700">#0097a7</color>
    <color name="md_cyan_800">#00838f</color>
    <color name="md_cyan_900">#006064</color>
    <color name="md_cyan_A100">#84ffff</color>
    <color name="md_cyan_A200">#18ffff</color>
    <color name="md_cyan_A400">#00e5ff</color>
    <color name="md_cyan_A700">#00b8d4</color>
    <color name="md_cyan_500_25">#4000bcd4</color>
    <color name="md_cyan_500_50">#8000bcd4</color>
    <color name="md_cyan_500_75">#c000bcd4</color>
    <!-- teals -->
    <array name="teals">
        <item>@color/md_teal_500</item>
        <item>@color/md_teal_50</item>
        <item>@color/md_teal_100</item>
        <item>@color/md_teal_200</item>
        <item>@color/md_teal_300</item>
        <item>@color/md_teal_400</item>
        <item>@color/md_teal_600</item>
        <item>@color/md_teal_700</item>
        <item>@color/md_teal_800</item>
        <item>@color/md_teal_900</item>
        <item>@color/md_teal_A100</item>
        <item>@color/md_teal_A200</item>
        <item>@color/md_teal_A400</item>
        <item>@color/md_teal_A700</item>
    </array>
    <color name="md_teal_50">#e0f2f1</color>
    <color name="md_teal_100">#b2dfdb</color>
    <color name="md_teal_200">#80cbc4</color>
    <color name="md_teal_300">#4db6ac</color>
    <color name="md_teal_400">#26a69a</color>
    <color name="md_teal_500">#009688</color>
    <color name="md_teal_600">#00897b</color>
    <color name="md_teal_700">#00796b</color>
    <color name="md_teal_800">#00695c</color>
    <color name="md_teal_900">#004d40</color>
    <color name="md_teal_A100">#a7ffeb</color>
    <color name="md_teal_A200">#64ffda</color>
    <color name="md_teal_A400">#1de9b6</color>
```

```
<color name="md_teal_A700">#00bfa5</color>
<color name="md_teal_500_25">#40009688</color>
<color name="md_teal_500_50">#80009688</color>
<color name="md_teal_500_75">#c8009688</color>
<!-- greens -->
<array name="greens">
    <item>@color/md_green_500</item>
    <item>@color/md_green_50</item>
    <item>@color/md_green_100</item>
    <item>@color/md_green_200</item>
    <item>@color/md_green_300</item>
    <item>@color/md_green_400</item>
    <item>@color/md_green_600</item>
    <item>@color/md_green_700</item>
    <item>@color/md_green_800</item>
    <item>@color/md_green_900</item>
    <item>@color/md_green_A100</item>
    <item>@color/md_green_A200</item>
    <item>@color/md_green_A400</item>
    <item>@color/md_green_A700</item>
</array>
<color name="md_green_50">#e8f5e9</color>
<color name="md_green_100">#c8e6c9</color>
<color name="md_green_200">#a5d6a7</color>
<color name="md_green_300">#81c784</color>
<color name="md_green_400">#66bb6a</color>
<color name="md_green_500">#4caf50</color>
<color name="md_green_600">#43a047</color>
<color name="md_green_700">#388e3c</color>
<color name="md_green_800">#2e7d32</color>
<color name="md_green_900">#1b5e20</color>
<color name="md_green_A100">#b9f6ca</color>
<color name="md_green_A200">#69f0ae</color>
<color name="md_green_A400">#00E676</color>
<color name="md_green_A700">#00c853</color>
<color name="md_green_500_25">#404caf50</color>
<color name="md_green_500_50">#804caf50</color>
<color name="md_green_500_75">#c04caf50</color>
<!--light greens-->
<array name="light_greens">
    <item>@color/md_light_green_500</item>
    <item>@color/md_light_green_50</item>
    <item>@color/md_light_green_100</item>
    <item>@color/md_light_green_200</item>
    <item>@color/md_light_green_300</item>
    <item>@color/md_light_green_400</item>
    <item>@color/md_light_green_600</item>
    <item>@color/md_light_green_700</item>
    <item>@color/md_light_green_800</item>
    <item>@color/md_light_green_900</item>
    <item>@color/md_light_green_A100</item>
    <item>@color/md_light_green_A200</item>
    <item>@color/md_light_green_A400</item>
    <item>@color/md_light_green_A700</item>
</array>
<color name="md_light_green_50">#f1f8e9</color>
<color name="md_light_green_100">#dcedc8</color>
<color name="md_light_green_200">#c5e1a5</color>
<color name="md_light_green_300">#aed581</color>
<color name="md_light_green_400">#9ccc65</color>
<color name="md_light_green_500">#8bc34a</color>
```

```
<color name="md_light_green_600">#7cb342</color>
<color name="md_light_green_700">#689f38</color>
<color name="md_light_green_800">#558b2f</color>
<color name="md_light_green_900">#33691e</color>
<color name="md_light_green_A100">#ccff90</color>
<color name="md_light_green_A200">#b2ff59</color>
<color name="md_light_green_A400">#76FF03</color>
<color name="md_light_green_A700">#64dd17</color>
<color name="md_light_green_500_25">#408bc34a</color>
<color name="md_light_green_500_50">#808bc34a</color>
<color name="md_light_green_500_75">#c88bc34a</color>
<!-- limes -->
<array name="limes">
    <item>@color/md_lime_500</item>
    <item>@color/md_lime_50</item>
    <item>@color/md_lime_100</item>
    <item>@color/md_lime_200</item>
    <item>@color/md_lime_300</item>
    <item>@color/md_lime_400</item>
    <item>@color/md_lime_600</item>
    <item>@color/md_lime_700</item>
    <item>@color/md_lime_800</item>
    <item>@color/md_lime_900</item>
    <item>@color/md_lime_A100</item>
    <item>@color/md_lime_A200</item>
    <item>@color/md_lime_A400</item>
    <item>@color/md_lime_A700</item>
</array>
<color name="md_lime_50">#f9fbe7</color>
<color name="md_lime_100">#f0f4c3</color>
<color name="md_lime_200">#e6ee9c</color>
<color name="md_lime_300">#dce775</color>
<color name="md_lime_400">#d4e157</color>
<color name="md_lime_500">#cddc39</color>
<color name="md_lime_600">#c0ca33</color>
<color name="md_lime_700">#afb42b</color>
<color name="md_lime_800">#9e9d24</color>
<color name="md_lime_900">#827717</color>
<color name="md_lime_A100">#f4ff81</color>
<color name="md_lime_A200">#eeff41</color>
<color name="md_lime_A400">#c6ff00</color>
<color name="md_lime_A700">#aeea00</color>
<color name="md_lime_500_25">#40cddc39</color>
<color name="md_lime_500_50">#80cddc39</color>
<color name="md_lime_500_75">#c0cddc39</color>
<!--yellows -->
<array name="yellows">
    <item>@color/md_yellow_500</item>
    <item>@color/md_yellow_50</item>
    <item>@color/md_yellow_100</item>
    <item>@color/md_yellow_200</item>
    <item>@color/md_yellow_300</item>
    <item>@color/md_yellow_400</item>
    <item>@color/md_yellow_600</item>
    <item>@color/md_yellow_700</item>
    <item>@color/md_yellow_800</item>
    <item>@color/md_yellow_900</item>
    <item>@color/md_yellow_A100</item>
    <item>@color/md_yellow_A200</item>
    <item>@color/md_yellow_A400</item>
    <item>@color/md_yellow_A700</item>
```

```
</array>
<color name="md_yellow_50">#fffde7</color>
<color name="md_yellow_100">#fff9c4</color>
<color name="md_yellow_200">#fff59d</color>
<color name="md_yellow_300">#fff176</color>
<color name="md_yellow_400">#ffee58</color>
<color name="md_yellow_500">#ffeb3b</color>
<color name="md_yellow_600">#fdd835</color>
<color name="md_yellow_700">#fbc02d</color>
<color name="md_yellow_800">#f9a825</color>
<color name="md_yellow_900">#f57f17</color>
<color name="md_yellow_A100">#ffff8d</color>
<color name="md_yellow_A200">#ffff00</color>
<color name="md_yellow_A400">#ffea00</color>
<color name="md_yellow_A700">#ffd600</color>
<color name="md_yellow_500_25">#40ffeb3b</color>
<color name="md_yellow_500_50">#80ffeb3b</color>
<color name="md_yellow_500_75">#c0ffeb3b</color>
<!-- ambers -->
<array name="ambers">
    <item>@color/md_amber_500</item>
    <item>@color/md_amber_50</item>
    <item>@color/md_amber_100</item>
    <item>@color/md_amber_200</item>
    <item>@color/md_amber_300</item>
    <item>@color/md_amber_400</item>
    <item>@color/md_amber_600</item>
    <item>@color/md_amber_700</item>
    <item>@color/md_amber_800</item>
    <item>@color/md_amber_900</item>
    <item>@color/md_amber_A100</item>
    <item>@color/md_amber_A200</item>
    <item>@color/md_amber_A400</item>
    <item>@color/md_amber_A700</item>
</array>
<color name="md_amber_50">#FFF8E1</color>
<color name="md_amber_100">#ffecb3</color>
<color name="md_amber_200">#FFE082</color>
<color name="md_amber_300">#ffd54f</color>
<color name="md_amber_400">#ffca28</color>
<color name="md_amber_500">#ffc107</color>
<color name="md_amber_600">#ffb300</color>
<color name="md_amber_700">#ffa000</color>
<color name="md_amber_800">#ff8f00</color>
<color name="md_amber_900">#ff6f00</color>
<color name="md_amber_A100">#ffe57f</color>
<color name="md_amber_A200">#ffd740</color>
<color name="md_amber_A400">#ffc400</color>
<color name="md_amber_A700">#ffab00</color>
<color name="md_amber_500_25">#40ffc107</color>
<color name="md_amber_500_50">#80ffc107</color>
<color name="md_amber_500_75">#c0ffc107</color>
<!-- oranges -->
<array name="oranges">
    <item>@color/md_orange_500</item>
    <item>@color/md_orange_50</item>
    <item>@color/md_orange_100</item>
    <item>@color/md_orange_200</item>
    <item>@color/md_orange_300</item>
    <item>@color/md_orange_400</item>
    <item>@color/md_orange_600</item>
```

```
        <item>@color/md_orange_700</item>
        <item>@color/md_orange_800</item>
        <item>@color/md_orange_900</item>
        <item>@color/md_orange_A100</item>
        <item>@color/md_orange_A200</item>
        <item>@color/md_orange_A400</item>
        <item>@color/md_orange_A700</item>
    </array>
    <color name="md_orange_50">#fff3e0</color>
    <color name="md_orange_100">#ffe0b2</color>
    <color name="md_orange_200">#ffcc80</color>
    <color name="md_orange_300">#ffb74d</color>
    <color name="md_orange_400">#ffa726</color>
    <color name="md_orange_500">#ff9800</color>
    <color name="md_orange_600">#fb8c00</color>
    <color name="md_orange_700">#f57c00</color>
    <color name="md_orange_800">#ef6c00</color>
    <color name="md_orange_900">#e65100</color>
    <color name="md_orange_A100">#ffd180</color>
    <color name="md_orange_A200">#ffab40</color>
    <color name="md_orange_A400">#ff9100</color>
    <color name="md_orange_A700">#ff6d00</color>
    <color name="md_orange_500_25">#40ff9800</color>
    <color name="md_orange_500_50">#80ff9800</color>
    <color name="md_orange_500_75">#c0ff9800</color>
    <!-- deep oranges -->
    <array name="deep_oranges">
        <item>@color/md_deep_orange_500</item>
        <item>@color/md_deep_orange_50</item>
        <item>@color/md_deep_orange_100</item>
        <item>@color/md_deep_orange_200</item>
        <item>@color/md_deep_orange_300</item>
        <item>@color/md_deep_orange_400</item>
        <item>@color/md_deep_orange_600</item>
        <item>@color/md_deep_orange_700</item>
        <item>@color/md_deep_orange_800</item>
        <item>@color/md_deep_orange_900</item>
        <item>@color/md_deep_orange_A100</item>
        <item>@color/md_deep_orange_A200</item>
        <item>@color/md_deep_orange_A400</item>
        <item>@color/md_deep_orange_A700</item>
    </array>
    <color name="md_deep_orange_50">#fbe9e7</color>
    <color name="md_deep_orange_100">#ffccbc</color>
    <color name="md_deep_orange_200">#ffab91</color>
    <color name="md_deep_orange_300">#ff8a65</color>
    <color name="md_deep_orange_400">#ff7043</color>
    <color name="md_deep_orange_500">#ff5722</color>
    <color name="md_deep_orange_600">#f4511e</color>
    <color name="md_deep_orange_700">#e64a19</color>
    <color name="md_deep_orange_800">#d84315</color>
    <color name="md_deep_orange_900">#bf360c</color>
    <color name="md_deep_orange_A100">#ff9e80</color>
    <color name="md_deep_orange_A200">#ff6e40</color>
    <color name="md_deep_orange_A400">#ff3d00</color>
    <color name="md_deep_orange_A700">#dd2c00</color>
    <color name="md_deep_orange_500_25">#40ff5722</color>
    <color name="md_deep_orange_500_50">#80ff5722</color>
    <color name="md_deep_orange_500_75">#c0ff5722</color>
    <!-- browns -->
    <array name="browns">
```

```
        <item>@color/md_brown_500</item>
        <item>@color/md_brown_50</item>
        <item>@color/md_brown_100</item>
        <item>@color/md_brown_200</item>
        <item>@color/md_brown_300</item>
        <item>@color/md_brown_400</item>
        <item>@color/md_brown_600</item>
        <item>@color/md_brown_700</item>
        <item>@color/md_brown_800</item>
        <item>@color/md_brown_900</item>
    </array>
    <color name="md_brown_50">#efebe9</color>
    <color name="md_brown_100">#d7ccc8</color>
    <color name="md_brown_200">#bcaaa4</color>
    <color name="md_brown_300">#a1887f</color>
    <color name="md_brown_400">#8d6e63</color>
    <color name="md_brown_500">#795548</color>
    <color name="md_brown_600">#6d4c41</color>
    <color name="md_brown_700">#5d4037</color>
    <color name="md_brown_800">#4e342e</color>
    <color name="md_brown_900">#3e2723</color>
    <color name="md_brown_500_25">#40795548</color>
    <color name="md_brown_500_50">#80795548</color>
    <color name="md_brown_500_75">#c0795548</color>
    <!--grey-->
    <array name="greys">
        <item>@color/md_grey_500</item>
        <item>@color/md_grey_50</item>
        <item>@color/md_grey_100</item>
        <item>@color/md_grey_200</item>
        <item>@color/md_grey_300</item>
        <item>@color/md_grey_400</item>
        <item>@color/md_grey_600</item>
        <item>@color/md_grey_700</item>
        <item>@color/md_grey_800</item>
        <item>@color/md_grey_900</item>
        <item>@color/md_black_1000</item>
        <item>@color/md_white_1000</item>
    </array>
    <color name="md_grey_50">#fafafa</color>
    <color name="md_grey_100">#f5f5f5</color>
    <color name="md_grey_200">#eeeeee</color>
    <color name="md_grey_300">#e0e0e0</color>
    <color name="md_grey_400">#bdbdbd</color>
    <color name="md_grey_500">#9e9e9e</color>
    <color name="md_grey_600">#757575</color>
    <color name="md_grey_700">#616161</color>
    <color name="md_grey_800">#424242</color>
    <color name="md_grey_900">#212121</color>
    <color name="md_grey_500_25">#409e9e9e</color>
    <color name="md_grey_500_50">#809e9e9e</color>
    <color name="md_grey_500_75">#c09e9e9e</color>
    <color name="md_white_1000">#FFFFFF</color>
    <color name="md_white_1000_10">#1affffff</color>
    <color name="md_white_1000_15">#22ffffff</color>
    <color name="md_white_1000_20">#33ffffff</color>
    <color name="md_white_1000_25">#40ffffff</color>
    <color name="md_white_1000_50">#80ffffff</color>
    <color name="md_white_1000_75">#c0ffffff</color>
    <color name="md_black_1000_10">#1a000000</color>
    <color name="md_black_1000_15">#26000000</color>
```

```
    <color name="md_black_1000_20">#33000000</color>
    <color name="md_black_1000_25">#40000000</color>
    <color name="md_black_1000_50">#80000000</color>
    <color name="md_black_1000_75">#c0000000</color>
    <color name="md_black_1000">#000000</color>
    <!--blue grey-->
    <array name="blue_greys">
        <item>@color/md_blue_grey_500</item>
        <item>@color/md_blue_grey_50</item>
        <item>@color/md_blue_grey_100</item>
        <item>@color/md_blue_grey_200</item>
        <item>@color/md_blue_grey_300</item>
        <item>@color/md_blue_grey_400</item>
        <item>@color/md_blue_grey_600</item>
        <item>@color/md_blue_grey_700</item>
        <item>@color/md_blue_grey_800</item>
        <item>@color/md_blue_grey_900</item>
    </array>
    <color name="md_blue_grey_50">#eceff1</color>
    <color name="md_blue_grey_100">#cfd8dc</color>
    <color name="md_blue_grey_200">#b0bec5</color>
    <color name="md_blue_grey_300">#90a4ae</color>
    <color name="md_blue_grey_400">#78909c</color>
    <color name="md_blue_grey_500">#607d8b</color>
    <color name="md_blue_grey_600">#546e7a</color>
    <color name="md_blue_grey_700">#455a64</color>
    <color name="md_blue_grey_800">#37474f</color>
    <color name="md_blue_grey_900">#263238</color>
    <color name="md_blue_grey_500_25">#40607d8b</color>
    <color name="md_blue_grey_500_50">#80607d8b</color>
    <color name="md_blue_grey_500_75">#c0607d8b</color>
    <!--Texts-->
    <array name="texts_white">
        <item>@color/md_text_white</item>
        <item>@color/md_secondary_text_icons_white</item>
        <item>@color/md_disabled_hint_text_white</item>
        <item>@color/md_divider_white</item>
    </array>
    <color name="md_text_white">#ffffffff</color>
    <color name="md_secondary_text_icons_white">#b3ffffff</color>
    <color name="md_disabled_hint_text_white">#4dffffff</color>
    <color name="md_divider_white">#1fffffff</color>
    <array name="texts_black">
        <item>@color/md_text</item>
        <item>@color/md_secondary_text_icons</item>
        <item>@color/md_disabled_hint_text</item>
        <item>@color/md_divider</item>
    </array>
    <color name="md_text">#df000000</color>
    <color name="md_secondary_text_icons">#8a000000</color>
    <color name="md_disabled_hint_text">#4c000000</color>
    <color name="md_divider">#1f000000</color>
</resources>
```

dimens.xml

```
<resources>

    <dimen name="activity_horizontal_margin">16dp</dimen>
    <dimen name="activity_vertical_margin">16dp</dimen>
    <dimen name="design_navigation_icon_size">40dp</dimen>
</resources>
```

styles.xml

```
<resources>

    <!-- Base application theme. -->
    <style name="AppTheme" parent="Theme.AppCompat.NoActionBar">
        <!-- Customize your theme here. -->
        <item name="colorPrimary">@color/md_black_1000_75</item>

    </style>
    <style name="Principal" parent="Theme.AppCompat">
        <item name="colorPrimary">@color/md_blue_500_75</item>
    </style>
    <style name="Transparentt" parent="Theme.AppCompat">
        <item name="colorPrimary">@android:color/transparent</item>
    </style>

</resources>
```

strings.xml

```
<resources>
    <string name="app_name">App Camaleon</string>
    <string name="app_camaleon">APP Camaleon</string>
    <string name="iniciar_sesi_n">Iniciar Sesión</string>
    <string name="primero">Primero</string>
    <string name="principal">Principal</string>
    <string name="tercero">Tercero</string>
    <string name="secundario">Secundario</string>
    <string name="title_activity_main2">Main2Activity</string>
    <string name="sensores">Sensores</string>
    <string name="visi_n">Visión</string>
    <string name="otros">Otros</string>
    <string name="temperatura">Temperatura:</string>
    <string name="calor">Calor:</string>
    <string name="modo_verano">Modo Verano</string>
    <string name="title_home">Home</string>
    <string name="title_dashboard">Dashboard</string>
    <string name="title_notifications">Notifications</string>
    <string name="opciones">Opciones</string>
    <string name="temperatura_m_xima">Temperatura Máxima:</string>
    <string name="title_activity_camara">Camara</string>
    <string name="title_activity_maps">Map</string>
    <string name="textview">TextView</string>
    <string name="conectado">Conectado</string>
    <string name="humedad">Humedad:</string>
    <string name="title_activity_login">Sign in</string>
    <!-- Strings related to login -->
    <string name="prompt_email">Email</string>
    <string name="prompt_password">Password (optional)</string>
    <string name="action_sign_in">Sign in or register</string>
    <string name="action_sign_in_short">Sign in</string>
    <string name="error_invalid_email">This email address is
invalid</string>
    <string name="error_invalid_password">This password is too
short</string>
    <string name="error_incorrect_password">This password is
incorrect</string>
    <string name="error_field_required">This field is
required</string>
    <string name="permission_rationale">"Contacts permissions are
needed for providing email
        completions."
    </string>
    <string name="iniciar_session">Iniciar Session</string>
    <string name="iniciar">Iniciar</string>
    <string name="registrarse">Registrarse</string>
    <string name="togglebutton">ToggleButton</string>
    <string name="apagado">Apagado</string>
    <string name="_00">00</string>
</resources>
```

Annex II: Arduino

```
#include <IRremote.h>
#include <IRremoteInt.h>
#include <SimpleDHT.h>
#include <SoftwareSerial.h>
#include <LiquidCrystal.h>
#include <Wire.h>
#include "RTClib.h"
#define TIMER_ENABLE_INTR    (TIMSK5 = _BV(OCIE5A))
#define TIMER_DISABLE_INTR   (TIMSK5 = 0)

LiquidCrystal lcd(43, 41, 39, 37, 35, 33);//(7, 8, 9, 10, 11, 12)
RTC_DS3231 rtc;
String Recive = "";
int Ventilador=4;
int Pantalla=5;
bool Inicio=false;
int Temp = 2; //Pin del sensor de temperatura
int Reset=12;
SimpleDHT11 dht11;
int Calor=52;
int Luz1=50;
int Luz2=48;
int EstadoLuces;
int temperatureCheck = 0;
int humidityCheck = 0;
byte temperature = 0;
byte humidity = 0;
String LCD,Fecha,setFecha,DatosRele,Tempp,Hora,setHoraDesp;
String setLuzDorm="2100";
String setLuz1Dorm="2100";
String setCalorDorm2="2300";
String setDesp2="0800";
unsigned int setDesp= 800;
unsigned int setCalorDorm=2300;
bool EstadoLCD=false;
bool inicio=false;
int Receptor = 3;
String Minutos,Horas;
IRrecv REC(Receptor);
decode_results RES;
```

```
void setup() {

Serial.begin(9600);//Baudios
Serial1.begin(9600);
Serial1.print("Modulo Conectado$");

REC.enableIRIn();
lcd.begin(16, 2);
lcd.print("App Camaleon");
delay(3000);

if (! rtc.begin()) {while (1);}
 if (rtc.lostPower()) { rtc.adjust(DateTime(F(__DATE__), F(__TIME__))); }

pinMode(Calor,OUTPUT);
pinMode(Luz1,OUTPUT)
pinMode(Luz2,OUTPUT);
pinMode(Reset,OUTPUT);
pinMode(Ventilador,OUTPUT);
pinMode(Pantalla,OUTPUT);

digitalWrite(Calor,HIGH);
digitalWrite(Luz1,HIGH);
digitalWrite(Luz2,LOW);
digitalWrite(Ventilador,HIGH);
digitalWrite(Pantalla,HIGH);

}

void loop() {

DateTime now = rtc.now();
if(now.minute()<10){Minutos="0"+(String)now.minute();}else{Minutos=(String)now.minute();
if(now.hour()<10){Horas="0"+(String)now.hour();}else{Horas=(String)now.hour();}
Fecha = (String)now.day() + "/" + (String)now.month() + "      " + Horas + ":" + Minutos + "   ";
setFecha=Horas+Minutos;

if(temperature > 30 && digitalRead(Calor)==HIGH && digitalRead(Ventilador)==HIGH){

   digitalWrite(Calor,LOW);
   digitalWrite(Luz1,HIGH);
   digitalWrite(Luz2,HIGH);
  }
```

```
if(temperature < 29 && digitalRead(Calor)==LOW && digitalRead(Ventilador)==HIGH){

   digitalWrite(Calor,HIGH);
   digitalWrite(Luz1,HIGH);
   digitalWrite(Luz2,LOW);
   }

LCD= "Temp: " + (String)temperature+" "+"Hum: " + (String)humidity;
lcd.setCursor(0, 0);
lcd.print(Fecha);
lcd.setCursor(0, 1);
lcd.print(LCD);

if(Hora.length()==4){
      setLuzDorm=Hora;
      setLuz1Dorm=Hora;
      setCalorDorm=setLuzDorm.toInt() + 200;
      setDesp= setLuzDorm.toInt() + 1100;

      if (setDesp > 2359){
          setDesp=setDesp-2400;

          if(String(setDesp).length()==3){setDesp2="0"+(String)setDesp;}
          else{setDesp2=(String)setDesp;}

      }else {setDesp2=(String)setDesp;}

     if (setCalorDorm > 2359){
        setCalorDorm=setCalorDorm-2400;

        switch(String(setCalorDorm).length())

        {
        case 4:
          setCalorDorm2=(String)setCalorDorm;
        break;
       case 3:
          setCalorDorm2="0"+(String)setCalorDorm;
       break;
       case 2:
          setCalorDorm2="00"+(String)setCalorDorm;
       break;
```

```
       case 1:
          setCalorDorm2="000"+(String)setCalorDorm;
        break;
        case 0:
          setCalorDorm2="0000";
        break;
     }

}else{
if(String(setCalorDorm).length()==3){setCalorDorm2="0"+(String)setCalorDorm;}
          else{setCalorDorm2=(String)setCalorDorm;}

            }
     Hora="";
  }

If (setLuzDorm.substring(0,2)==setFecha.substring(0,2) &&
setLuzDorm.substring(2,4)=                                    =setFecha.substring(2,4)
&&( digitalRead(Luz1)==HIGH || digitalRead(Luz2)==HIGH) )
{
   digitalWrite(Luz1,LOW);
   digitalWrite(Luz2,LOW);
}

if(setCalorDorm2.substring(0,2)==setFecha.substring(0,2) &&
setCalorDorm2.substring(2,4)= =setFecha.substring(2,4) &&
digitalRead(Calor)==HIGH )
 {
    digitalWrite(Calor,LOW);
    digitalWrite(Ventilador,LOW);
    digitalWrite(Pantalla,LOW);
}

if(setDesp2.substring(0,2)==setFecha.substring(0,2) && setDesp2.substring(2,4)=
=setFecha.substring(2,4)&&(digitalRead(Calor)==LOW || digitalRead(Luz1)==LOW) )
{
   digitalWrite(Calor,HIGH);
   digitalWrite(Luz1,HIGH);
   digitalWrite(Ventilador,HIGH);
   digitalWrite(Pantalla,HIGH);
}

TIMER_DISABLE_INTR ;
dht11.read(Temp, &temperature, &humidity, NULL);
TIMER_ENABLE_INTR ;
delay(300);
```

```
if (REC.decode(&RES))
{
  switch(RES.value)
  {
   case 0xFFA25D: //POWER
          if (EstadoLCD==true){
             lcd.noDisplay();
             EstadoLCD=false;
             digitalWrite(Pantalla,LOW);
          }else{
            lcd.display();
            EstadoLCD=true;
            digitalWrite(Pantalla,HIGH);
          }
  break;

  case 0xFF02FD: //Pause
           if(digitalRead(Ventilador)==HIGH){digitalWrite(Ventilador,LOW);}
          else{digitalWrite(Ventilador,HIGH);}
  break;
  case 0xFF9867:  //EQ
          lcd.setCursor(0, 1);
         LCD="Encendido: "+setDesp2.substring(0,2)+":"+setDesp2.substring(2,4);
          lcd.print(LCD);
         delay(1000);

  break;

  case 0xFF6897: //0
         digitalWrite(Calor,HIGH);
        digitalWrite(Luz1,HIGH);
        digitalWrite(Luz2,LOW);
        setDesp2= "0800";
        setCalorDorm2="2300";


        setLuzDorm="2100";
        setLuz1Dorm="2100";
        Serial1.print(setLuzDorm+"$");

  break;
```

```
 case 0xFF30CF: //1
        if(digitalRead(Calor)==HIGH){digitalWrite(Calor,LOW); }
        else{digitalWrite(Calor,HIGH);}

  break;

 case 0xFF18E7: //2
         if(digitalRead(Luz1)==HIGH){digitalWrite(Luz1,LOW);}
        else{digitalWrite(Luz1,HIGH);}
 break;

case 0xFF7A85: //3
           if(digitalRead(Luz2)==HIGH){digitalWrite(Luz2,LOW);}
           else{digitalWrite(Luz2,HIGH);}
break;

case 0xFF10EF: //4
            lcd.setCursor(0, 1);
            LCD="Es: "+(String)digitalRead(Calor)+"   H:
"+setCalorDorm2.substring(0,2)  +":"
            +  setCalorDorm2.substring(2,4);
           lcd.print(LCD);
           delay(1000);
break;

case 0xFF38C7: //5
          lcd.setCursor(0, 1);
          LCD="Es:"+(String)digitalRead(Luz1)+" H:
"+String(setLuzDorm).substring(0,2) +":"
          +setLuzDorm.substring(2,4);
          lcd.print(LCD);
          delay(1000);

break;

case 0xFF5AA5: //6
        lcd.setCursor(0, 1);
        LCD="Es: "+(String)digitalRead(Luz2)+"   H: " + setLuz1Dorm.substring(0,2)+
":" +
        setLuz1Dorm.substring(2,4);

       lcd.print(LCD);
       delay(1000);
break;
case 0xFFE21D:  break;//Func/Stop
case 0xFF629D:  break; //Serial.println("VOL+");
```

```
case 0xFF22DD:  break;//FastBack
case 0xFFC23D:  break; //FastFordward
case 0xFFE01F:  break;//Serial.println("DOWN");

case 0xFFA857:  break; //VOL-
case 0xFF906F:  break;//UP
case 0xFF42BD:  break; //7
case 0xFF4AB5:  break; //8
case 0xFF52AD:  break; //9

default:
        lcd.setCursor(4, 0);
        lcd.print("XXXX");

}
delay(100);
REC.resume();

}

if (Serial1.available())
 {
 char Rec;
 Rec = Serial1.read();
 Serial.print(Rec);
      switch (Rec) {

case 'T':

          delay(100);
          if(digitalRead(Calor)==HIGH){EstadoLuces= EstadoLuces + 1; }
          if(digitalRead(Luz1)==HIGH) {EstadoLuces= EstadoLuces + 2; }
         if(digitalRead(Luz2)==HIGH) {EstadoLuces= EstadoLuces + 4;}
         DatosRele="00"+(String)EstadoLuces;
         Tempp=DatosRele+(String)humidity+(String)temperature +
         (String)digitalRead(Ventilador)+ "#";
         Serial1.print(Tempp);
         EstadoLuces=0;

break;

case 'c':
       delay(100);
       Serial1.print(setLuzDorm+"$");
break;
```

```
case 'Q':
        if(digitalRead(Calor)==HIGH){digitalWrite(Calor,LOW);}
        else{digitalWrite(Calor,HIGH);}
break;

case 'W':
          if(digitalRead(Luz1)==HIGH){digitalWrite(Luz1,LOW);}
          else{digitalWrite(Luz1,HIGH);}

break;

case 'R':
          if(digitalRead(Luz2)==HIGH){digitalWrite(Luz2,LOW);}
          else{digitalWrite(Luz2,HIGH);}

break;

case 'V':
        if(digitalRead(Ventilador)==HIGH){digitalWrite(Ventilador,LOW);}
        else{digitalWrite(Ventilador,HIGH);}

break;
case'0':
       Hora=Hora+"0";
break;
case'1':
       Hora=Hora+"1" ;
break;
case'2':
        Hora=Hora+"2";
break;
case'3':
         Hora=Hora+"3";
 break;
 case'4':
        Hora=Hora+"4";
break;
case'5':
 Hora=Hora+"5";
break;
 case'6':
 Hora=Hora+"6";
break;
 case'7':
 Hora=Hora+"7";
break;
```

```
case'8':
 Hora=Hora+"8";
break;


case'9':
Hora=Hora+"9";
break;
}
}
```

IRremote.h

```
#ifndef IRremote_h
#define IRremote_h

// The following are compile-time library options.
// If you change them, recompile the library.
// If DEBUG is defined, a lot of debugging output will be
printed during decoding.
// TEST must be defined for the IRtest unittests to work.  It
will make some
// methods virtual, which will be slightly slower, which is why
it is optional.
// #define DEBUG
// #define TEST

// Results returned from the decoder
class decode_results {
public:
  int decode_type; // NEC, SONY, RC5, UNKNOWN
  union { // This is used for decoding Panasonic and Sharp data
    unsigned int panasonicAddress;
    unsigned int sharpAddress;
  };
  unsigned long value; // Decoded value
  int bits; // Number of bits in decoded value
  volatile unsigned int *rawbuf; // Raw intervals in .5 us ticks
  int rawlen; // Number of records in rawbuf.
};

// Values for decode_type
#define NEC 1
#define SONY 2
#define RC5 3
#define RC6 4
#define DISH 5
#define SHARP 6
#define PANASONIC 7
#define JVC 8
#define SANYO 9
#define MITSUBISHI 10
#define SAMSUNG 11
#define LG 12
#define UNKNOWN -1

// Decoded value for NEC when a repeat code is received
#define REPEAT 0xffffffff

// main class for receiving IR
class IRrecv
{
public:
  IRrecv(int recvpin);
  void blink13(int blinkflag);
  int decode(decode_results *results);
```

```
  void enableIRIn();
  void resume();
private:
  // These are called by decode
  int getRClevel(decode_results *results, int *offset, int
*used, int t1);
  long decodeNEC(decode_results *results);
  long decodeSony(decode_results *results);
  long decodeSanyo(decode_results *results);
  long decodeMitsubishi(decode_results *results);
  long decodeRC5(decode_results *results);
  long decodeRC6(decode_results *results);
  long decodePanasonic(decode_results *results);
  long decodeLG(decode_results *results);
  long decodeJVC(decode_results *results);
  long decodeSAMSUNG(decode_results *results);
  long decodeHash(decode_results *results);
  int compare(unsigned int oldval, unsigned int newval);

}
;

// Only used for testing; can remove virtual for shorter code
#ifdef TEST
#define VIRTUAL virtual
#else
#define VIRTUAL
#endif

class IRsend
{
public:
  IRsend() {}
  void sendNEC(unsigned long data, int nbits);
  void sendSony(unsigned long data, int nbits);
  // Neither Sanyo nor Mitsubishi send is implemented yet
  //  void sendSanyo(unsigned long data, int nbits);
  //  void sendMitsubishi(unsigned long data, int nbits);
  void sendRaw(unsigned int buf[], int len, int hz);
  void sendRC5(unsigned long data, int nbits);
  void sendRC6(unsigned long data, int nbits);
  void sendDISH(unsigned long data, int nbits);
  void sendSharp(unsigned int address, unsigned int command);
  void sendSharpRaw(unsigned long data, int nbits);
  void sendPanasonic(unsigned int address, unsigned long data);
  void sendJVC(unsigned long data, int nbits, int repeat); //
*Note instead of sending the REPEAT constant if you want the JVC
repeat signal sent, send the original code value and change the
repeat argument from 0 to 1. JVC protocol repeats by skipping
the header NOT by sending a separate code value like NEC does.




// private:
```

```
  void sendSAMSUNG(unsigned long data, int nbits);
  void enableIROut(int khz);
  VIRTUAL void mark(int usec);
  VIRTUAL void space(int usec);
}
;

// Some useful constants

#define USECPERTICK 50  // microseconds per clock interrupt tick
#define RAWBUF 100 // Length of raw duration buffer

// Marks tend to be 100us too long, and spaces 100us too short
// when received due to sensor lag.
#define MARK_EXCESS 100

#endif
```

IRremoteInt.h

```
#ifndef IRremoteint_h
#define IRremoteint_h

#if defined(ARDUINO) && ARDUINO >= 100
#include <Arduino.h>
#else
#include <WProgram.h>
#endif

// define which timer to use
//
// Uncomment the timer you wish to use on your board.  If you
// are using another library which uses timer2, you have options
// to switch IRremote to use a different timer.

// Arduino Mega
#if defined(__AVR_ATmega1280__) || defined(__AVR_ATmega2560__)
  //#define IR_USE_TIMER1   // tx = pin 11
  #define IR_USE_TIMER2     // tx = pin 9
  //#define IR_USE_TIMER3   // tx = pin 5
  //#define IR_USE_TIMER4   // tx = pin 6
  //#define IR_USE_TIMER5   // tx = pin 46

// Teensy 1.0
#elif defined(__AVR_AT90USB162__)
  #define IR_USE_TIMER1     // tx = pin 17

// Teensy 2.0
#elif defined(__AVR_ATmega32U4__)
  //#define IR_USE_TIMER1   // tx = pin 14
  //#define IR_USE_TIMER3   // tx = pin 9
  #define IR_USE_TIMER4_HS  // tx = pin 10

// Teensy 3.0
#elif defined(__MK20DX128__)
  #define IR_USE_TIMER_CMT  // tx = pin 5

// Teensy++ 1.0 & 2.0
#elif defined(__AVR_AT90USB646__) ||
defined(__AVR_AT90USB1286__)
  //#define IR_USE_TIMER1   // tx = pin 25
  #define IR_USE_TIMER2     // tx = pin 1
  //#define IR_USE_TIMER3   // tx = pin 16

// Sanguino
#elif defined(__AVR_ATmega644P__) || defined(__AVR_ATmega644__)
  //#define IR_USE_TIMER1   // tx = pin 13
  #define IR_USE_TIMER2     // tx = pin 14

// Atmega8
#elif defined(__AVR_ATmega8P__) || defined(__AVR_ATmega8__)
  #define IR_USE_TIMER1   // tx = pin 9
```

```
// Arduino Duemilanove, Diecimila, LilyPad, Mini, Fio, etc
#else
  //#define IR_USE_TIMER1   // tx = pin 9
  #define IR_USE_TIMER2     // tx = pin 3
#endif



#ifdef F_CPU
#define SYSCLOCK F_CPU     // main Arduino clock
#else
#define SYSCLOCK 16000000  // main Arduino clock
#endif

#define ERR 0
#define DECODED 1


// defines for setting and clearing register bits
#ifndef cbi
#define cbi(sfr, bit) (_SFR_BYTE(sfr) &= ~_BV(bit))
#endif
#ifndef sbi
#define sbi(sfr, bit) (_SFR_BYTE(sfr) |= _BV(bit))
#endif

// Pulse parms are *50-100 for the Mark and *50+100 for the
space
// First MARK is the one after the long gap
// pulse parameters in usec
#define NEC_HDR_MARK  9000
#define NEC_HDR_SPACE 4500
#define NEC_BIT_MARK  560
#define NEC_ONE_SPACE 1600
#define NEC_ZERO_SPACE     560
#define NEC_RPT_SPACE 2250

#define SONY_HDR_MARK 2400
#define SONY_HDR_SPACE     600
#define SONY_ONE_MARK 1200
#define SONY_ZERO_MARK     600
#define SONY_RPT_LENGTH 45000
#define SONY_DOUBLE_SPACE_USECS  500  // usually ssee 713 - not
using ticks as get number wraparound

// SA 8650B
#define SANYO_HDR_MARK     3500  // seen range 3500
#define SANYO_HDR_SPACE    950 //  seen 950
#define SANYO_ONE_MARK     2400 // seen 2400
#define SANYO_ZERO_MARK 700 //  seen 700
#define SANYO_DOUBLE_SPACE_USECS  800  // usually ssee 713 - not
using ticks as get number wraparound
#define SANYO_RPT_LENGTH 45000

// Mitsubishi RM 75501
```

```
// 14200 7 41 7 42 7 42 7 17 7 17 7 18 7 41 7 18 7 17 7 17 7 18
7 41 8 17 7 17 7 18 7 17 7

// #define MITSUBISHI_HDR_MARK  250  // seen range 3500
#define MITSUBISHI_HDR_SPACE    350 //  7*50+100
#define MITSUBISHI_ONE_MARK     1950 // 41*50-100
#define MITSUBISHI_ZERO_MARK  750 // 17*50-100
// #define MITSUBISHI_DOUBLE_SPACE_USECS  800  // usually ssee
713 - not using ticks as get number wraparound
// #define MITSUBISHI_RPT_LENGTH 45000


#define RC5_T1        889
#define RC5_RPT_LENGTH     46000

#define RC6_HDR_MARK  2666
#define RC6_HDR_SPACE 889
#define RC6_T1        444
#define RC6_RPT_LENGTH     46000

#define SHARP_BIT_MARK 245
#define SHARP_ONE_SPACE 1805
#define SHARP_ZERO_SPACE 795
#define SHARP_GAP 600000
#define SHARP_TOGGLE_MASK 0x3FF
#define SHARP_RPT_SPACE 3000

#define DISH_HDR_MARK 400
#define DISH_HDR_SPACE 6100
#define DISH_BIT_MARK 400
#define DISH_ONE_SPACE 1700
#define DISH_ZERO_SPACE 2800
#define DISH_RPT_SPACE 6200
#define DISH_TOP_BIT 0x8000

#define PANASONIC_HDR_MARK 3502
#define PANASONIC_HDR_SPACE 1750
#define PANASONIC_BIT_MARK 502
#define PANASONIC_ONE_SPACE 1244
#define PANASONIC_ZERO_SPACE 400

#define JVC_HDR_MARK 8000
#define JVC_HDR_SPACE 4000
#define JVC_BIT_MARK 600
#define JVC_ONE_SPACE 1600
#define JVC_ZERO_SPACE 550
#define JVC_RPT_LENGTH 60000

#define LG_HDR_MARK 8000
#define LG_HDR_SPACE 4000
#define LG_BIT_MARK 600
#define LG_ONE_SPACE 1600
#define LG_ZERO_SPACE 550
#define LG_RPT_LENGTH 60000

#define SAMSUNG_HDR_MARK  5000
```

```
#define SAMSUNG_HDR_SPACE 5000
#define SAMSUNG_BIT_MARK  560
#define SAMSUNG_ONE_SPACE 1600
#define SAMSUNG_ZERO_SPACE  560
#define SAMSUNG_RPT_SPACE 2250


#define SHARP_BITS 15
#define DISH_BITS 16

#define TOLERANCE 25  // percent tolerance in measurements
#define LTOL (1.0 - TOLERANCE/100.)
#define UTOL (1.0 + TOLERANCE/100.)

#define _GAP 5000 // Minimum map between transmissions
#define GAP_TICKS (_GAP/USECPERTICK)

#define TICKS_LOW(us) (int) (((us)*LTOL/USECPERTICK))
#define TICKS_HIGH(us) (int) (((us)*UTOL/USECPERTICK + 1))

// receiver states
#define STATE_IDLE     2
#define STATE_MARK     3
#define STATE_SPACE    4
#define STATE_STOP     5

// information for the interrupt handler
typedef struct {
  uint8_t recvpin;           // pin for IR data from detector
  uint8_t rcvstate;          // state machine
  uint8_t blinkflag;         // TRUE to enable blinking of pin
13 on IR processing
  unsigned int timer;     // state timer, counts 50uS ticks.
  unsigned int rawbuf[RAWBUF]; // raw data
  uint8_t rawlen;         // counter of entries in rawbuf
}
irparams_t;

// Defined in IRremote.cpp
extern volatile irparams_t irparams;

// IR detector output is active low
#define MARK  0
#define SPACE 1

#define TOPBIT 0x80000000

#define NEC_BITS 32
#define SONY_BITS 12
#define SANYO_BITS 12
#define MITSUBISHI_BITS 16
#define MIN_RC5_SAMPLES 11
#define MIN_RC6_SAMPLES 1
#define PANASONIC_BITS 48
#define JVC_BITS 16
#define LG_BITS 28
```

```
#define SAMSUNG_BITS 32



// defines for timer2 (8 bits)
#if defined(IR_USE_TIMER2)
#define TIMER_RESET
#define TIMER_ENABLE_PWM     (TCCR2A |= _BV(COM2B1))
#define TIMER_DISABLE_PWM    (TCCR2A &= ~(_BV(COM2B1)))
#define TIMER_ENABLE_INTR    (TIMSK2 = _BV(OCIE2A))
#define TIMER_DISABLE_INTR   (TIMSK2 = 0)
#define TIMER_INTR_NAME      TIMER2_COMPA_vect
#define TIMER_CONFIG_KHZ(val) ({ \
  const uint8_t pwmval = SYSCLOCK / 2000 / (val); \
  TCCR2A = _BV(WGM20); \
  TCCR2B = _BV(WGM22) | _BV(CS20); \
  OCR2A = pwmval; \
  OCR2B = pwmval / 3; \
})
#define TIMER_COUNT_TOP      (SYSCLOCK * USECPERTICK / 1000000)
#if (TIMER_COUNT_TOP < 256)
#define TIMER_CONFIG_NORMAL() ({ \
  TCCR2A = _BV(WGM21); \
  TCCR2B = _BV(CS20); \
  OCR2A = TIMER_COUNT_TOP; \
  TCNT2 = 0; \
})
#else
#define TIMER_CONFIG_NORMAL() ({ \
  TCCR2A = _BV(WGM21); \
  TCCR2B = _BV(CS21); \
  OCR2A = TIMER_COUNT_TOP / 8; \
  TCNT2 = 0; \
})
#endif
#if defined(CORE_OC2B_PIN)
#define TIMER_PWM_PIN        CORE_OC2B_PIN  /* Teensy */
#elif defined(__AVR_ATmega1280__) || defined(__AVR_ATmega2560__)
#define TIMER_PWM_PIN        9  /* Arduino Mega */
#elif defined(__AVR_ATmega644P__) || defined(__AVR_ATmega644__)
#define TIMER_PWM_PIN        14 /* Sanguino */
#else
#define TIMER_PWM_PIN        3  /* Arduino Duemilanove,
Diecimila, LilyPad, etc */
#endif


// defines for timer1 (16 bits)
#elif defined(IR_USE_TIMER1)
#define TIMER_RESET
#define TIMER_ENABLE_PWM     (TCCR1A |= _BV(COM1A1))
#define TIMER_DISABLE_PWM    (TCCR1A &= ~(_BV(COM1A1)))
#if defined(__AVR_ATmega8P__) || defined(__AVR_ATmega8__)
  #define TIMER_ENABLE_INTR    (TIMSK = _BV(OCIE1A))
  #define TIMER_DISABLE_INTR   (TIMSK = 0)
```

```
#else
  #define TIMER_ENABLE_INTR     (TIMSK1 = _BV(OCIE1A))
  #define TIMER_DISABLE_INTR    (TIMSK1 = 0)
#endif
#define TIMER_INTR_NAME       TIMER1_COMPA_vect
#define TIMER_CONFIG_KHZ(val) ({ \
  const uint16_t pwmval = SYSCLOCK / 2000 / (val); \
  TCCR1A = _BV(WGM11); \
  TCCR1B = _BV(WGM13) | _BV(CS10); \
  ICR1 = pwmval; \
  OCR1A = pwmval / 3; \
})
#define TIMER_CONFIG_NORMAL() ({ \
  TCCR1A = 0; \
  TCCR1B = _BV(WGM12) | _BV(CS10); \
  OCR1A = SYSCLOCK * USECPERTICK / 1000000; \
  TCNT1 = 0; \
})
#if defined(CORE_OC1A_PIN)
#define TIMER_PWM_PIN         CORE_OC1A_PIN  /* Teensy */
#elif defined(__AVR_ATmega1280__) || defined(__AVR_ATmega2560__)
#define TIMER_PWM_PIN         11  /* Arduino Mega */
#elif defined(__AVR_ATmega644P__) || defined(__AVR_ATmega644__)
#define TIMER_PWM_PIN         13 /* Sanguino */
#else
#define TIMER_PWM_PIN         9  /* Arduino Duemilanove,
Diecimila, LilyPad, etc */
#endif


// defines for timer3 (16 bits)
#elif defined(IR_USE_TIMER3)
#define TIMER_RESET
#define TIMER_ENABLE_PWM      (TCCR3A |= _BV(COM3A1))
#define TIMER_DISABLE_PWM     (TCCR3A &= ~(_BV(COM3A1)))
#define TIMER_ENABLE_INTR     (TIMSK3 = _BV(OCIE3A))
#define TIMER_DISABLE_INTR    (TIMSK3 = 0)
#define TIMER_INTR_NAME       TIMER3_COMPA_vect
#define TIMER_CONFIG_KHZ(val) ({ \
  const uint16_t pwmval = SYSCLOCK / 2000 / (val); \
  TCCR3A = _BV(WGM31); \
  TCCR3B = _BV(WGM33) | _BV(CS30); \
  ICR3 = pwmval; \
  OCR3A = pwmval / 3; \
})
#define TIMER_CONFIG_NORMAL() ({ \
  TCCR3A = 0; \
  TCCR3B = _BV(WGM32) | _BV(CS30); \
  OCR3A = SYSCLOCK * USECPERTICK / 1000000; \
  TCNT3 = 0; \
})
#if defined(CORE_OC3A_PIN)
#define TIMER_PWM_PIN         CORE_OC3A_PIN  /* Teensy */
#elif defined(__AVR_ATmega1280__) || defined(__AVR_ATmega2560__)
#define TIMER_PWM_PIN         5  /* Arduino Mega */
#else
```

```
#error "Please add OC3A pin number here\n"
#endif


// defines for timer4 (10 bits, high speed option)
#elif defined(IR_USE_TIMER4_HS)
#define TIMER_RESET
#define TIMER_ENABLE_PWM     (TCCR4A |= _BV(COM4A1))
#define TIMER_DISABLE_PWM    (TCCR4A &= ~(_BV(COM4A1)))
#define TIMER_ENABLE_INTR    (TIMSK4 = _BV(TOIE4))
#define TIMER_DISABLE_INTR   (TIMSK4 = 0)
#define TIMER_INTR_NAME      TIMER4_OVF_vect
#define TIMER_CONFIG_KHZ(val) ({ \
  const uint16_t pwmval = SYSCLOCK / 2000 / (val); \
  TCCR4A = (1<<PWM4A); \
  TCCR4B = _BV(CS40); \
  TCCR4C = 0; \
  TCCR4D = (1<<WGM40); \
  TCCR4E = 0; \
  TC4H = pwmval >> 8; \
  OCR4C = pwmval; \
  TC4H = (pwmval / 3) >> 8; \
  OCR4A = (pwmval / 3) & 255; \
})
#define TIMER_CONFIG_NORMAL() ({ \
  TCCR4A = 0; \
  TCCR4B = _BV(CS40); \
  TCCR4C = 0; \
  TCCR4D = 0; \
  TCCR4E = 0; \
  TC4H = (SYSCLOCK * USECPERTICK / 1000000) >> 8; \
  OCR4C = (SYSCLOCK * USECPERTICK / 1000000) & 255; \
  TC4H = 0; \
  TCNT4 = 0; \
})
#if defined(CORE_OC4A_PIN)
#define TIMER_PWM_PIN        CORE_OC4A_PIN  /* Teensy */
#elif defined(__AVR_ATmega32U4__)
#define TIMER_PWM_PIN        13  /* Leonardo */
#else
#error "Please add OC4A pin number here\n"
#endif


// defines for timer4 (16 bits)
#elif defined(IR_USE_TIMER4)
#define TIMER_RESET
#define TIMER_ENABLE_PWM     (TCCR4A |= _BV(COM4A1))
#define TIMER_DISABLE_PWM    (TCCR4A &= ~(_BV(COM4A1)))
#define TIMER_ENABLE_INTR    (TIMSK4 = _BV(OCIE4A))
#define TIMER_DISABLE_INTR   (TIMSK4 = 0)
#define TIMER_INTR_NAME      TIMER4_COMPA_vect
#define TIMER_CONFIG_KHZ(val) ({ \
  const uint16_t pwmval = SYSCLOCK / 2000 / (val); \
  TCCR4A = _BV(WGM41); \
  TCCR4B = _BV(WGM43) | _BV(CS40); \
```

```
  ICR4 = pwmval; \
  OCR4A = pwmval / 3; \
})
#define TIMER_CONFIG_NORMAL() ({ \
  TCCR4A = 0; \
  TCCR4B = _BV(WGM42) | _BV(CS40); \
  OCR4A = SYSCLOCK * USECPERTICK / 1000000; \
  TCNT4 = 0; \
})
#if defined(CORE_OC4A_PIN)
#define TIMER_PWM_PIN        CORE_OC4A_PIN
#elif defined(__AVR_ATmega1280__) || defined(__AVR_ATmega2560__)
#define TIMER_PWM_PIN        6  /* Arduino Mega */
#else
#error "Please add OC4A pin number here\n"
#endif


// defines for timer5 (16 bits)
#elif defined(IR_USE_TIMER5)
#define TIMER_RESET
#define TIMER_ENABLE_PWM     (TCCR5A |= _BV(COM5A1))
#define TIMER_DISABLE_PWM    (TCCR5A &= ~(_BV(COM5A1)))
#define TIMER_ENABLE_INTR    (TIMSK5 = _BV(OCIE5A))
#define TIMER_DISABLE_INTR   (TIMSK5 = 0)
#define TIMER_INTR_NAME      TIMER5_COMPA_vect
#define TIMER_CONFIG_KHZ(val) ({ \
  const uint16_t pwmval = SYSCLOCK / 2000 / (val); \
  TCCR5A = _BV(WGM51); \
  TCCR5B = _BV(WGM53) | _BV(CS50); \
  ICR5 = pwmval; \
  OCR5A = pwmval / 3; \
})
#define TIMER_CONFIG_NORMAL() ({ \
  TCCR5A = 0; \
  TCCR5B = _BV(WGM52) | _BV(CS50); \
  OCR5A = SYSCLOCK * USECPERTICK / 1000000; \
  TCNT5 = 0; \
})
#if defined(CORE_OC5A_PIN)
#define TIMER_PWM_PIN        CORE_OC5A_PIN
#elif defined(__AVR_ATmega1280__) || defined(__AVR_ATmega2560__)
#define TIMER_PWM_PIN        46  /* Arduino Mega */
#else
#error "Please add OC5A pin number here\n"
#endif


// defines for special carrier modulator timer
#elif defined(IR_USE_TIMER_CMT)
#define TIMER_RESET ({                \
      uint8_t tmp = CMT_MSC;          \
      CMT_CMD2 = 30;             \
})
#define TIMER_ENABLE_PWM     CORE_PIN5_CONFIG =
PORT_PCR_MUX(2)|PORT_PCR_DSE|PORT_PCR_SRE
```

```
#define TIMER_DISABLE_PWM    CORE_PIN5_CONFIG =
PORT_PCR_MUX(1)|PORT_PCR_DSE|PORT_PCR_SRE
#define TIMER_ENABLE_INTR    NVIC_ENABLE_IRQ(IRQ_CMT)
#define TIMER_DISABLE_INTR   NVIC_DISABLE_IRQ(IRQ_CMT)
#define TIMER_INTR_NAME      cmt_isr
#ifdef ISR
#undef ISR
#endif
#define ISR(f) void f(void)
#if F_BUS == 48000000
#define CMT_PPS_VAL 5
#else
#define CMT_PPS_VAL 2
#endif
#define TIMER_CONFIG_KHZ(val) ({      \
      SIM_SCGC4 |= SIM_SCGC4_CMT;     \
      SIM_SOPT2 |= SIM_SOPT2_PTD7PAD; \
      CMT_PPS = CMT_PPS_VAL;          \
      CMT_CGH1 = 2667 / val;          \
      CMT_CGL1 = 5333 / val;          \
      CMT_CMD1 = 0;              \
      CMT_CMD2 = 30;             \
      CMT_CMD3 = 0;              \
      CMT_CMD4 = 0;              \
      CMT_OC = 0x60;             \
      CMT_MSC = 0x01;            \
})
#define TIMER_CONFIG_NORMAL() ({\
      SIM_SCGC4 |= SIM_SCGC4_CMT;     \
      CMT_PPS = CMT_PPS_VAL;          \
      CMT_CGH1 = 1;              \
      CMT_CGL1 = 1;              \
      CMT_CMD1 = 0;              \
      CMT_CMD2 = 30;             \
      CMT_CMD3 = 0;              \
      CMT_CMD4 = 19;             \
      CMT_OC = 0;                \
      CMT_MSC = 0x03;            \
})
#define TIMER_PWM_PIN        5


#else // unknown timer
#error "Internal code configuration error, no known
IR_USE_TIMER# defined\n"
#endif



// defines for blinking the LED
#if defined(CORE_LED0_PIN)
#define BLINKLED       CORE_LED0_PIN
#define BLINKLED_ON()  (digitalWrite(CORE_LED0_PIN, HIGH))
#define BLINKLED_OFF() (digitalWrite(CORE_LED0_PIN, LOW))
#elif defined(__AVR_ATmega1280__) || defined(__AVR_ATmega2560__)
#define BLINKLED       13
```

```
#define BLINKLED_ON()   (PORTB |= B10000000)
#define BLINKLED_OFF()  (PORTB &= B01111111)
#elif defined(__AVR_ATmega644P__) || defined(__AVR_ATmega644__)
#define BLINKLED        0
#define BLINKLED_ON()   (PORTD |= B00000001)
#define BLINKLED_OFF()  (PORTD &= B11111110)
#else
#define BLINKLED        13
#define BLINKLED_ON()   (PORTB |= B00100000)
#define BLINKLED_OFF()  (PORTB &= B11011111)
#endif

#endif
```

SimpleDHT.h

```
#ifndef __SIMPLE_DHT_H
#define __SIMPLE_DHT_H

#include <Arduino.h>
class SimpleDHT11 {
public:
    // to read from dht11.
    // @param pin the DHT11 pin.
    // @param ptemperature output, NULL to igore.
    // @param phumidity output, NULL to ignore.
    // @param pdata output 40bits sample, NULL to ignore.
    // @remark the min delay for this method is 1s.
    int read(int pin, byte* ptemperature, byte* phumidity, byte
pdata[40]);
private:
    // confirm the OUTPUT is level in us,
    // for example, when DHT11 start sample, it will
    //    1. PULL LOW 80us, call confirm(pin, 80, LOW)
    //    2. PULL HIGH 80us, call confirm(pin, 80, HIGH)
    // @return 0 success; oterwise, error.
    // @remark should never used to read bits,
    //    for function call use more time, maybe never got bit0.
    // @remark please use simple_dht11_read().
    int confirm(int pin, int us, byte level);
    // @data the bits of a byte.
    // @remark please use simple_dht11_read().
    byte bits2byte(byte data[8]);
    // read temperature and humidity from dht11.
    // @param pin the pin for DHT11, for example, 2.
    // @param data a byte[40] to read bits to 5bytes.
    // @return 0 success; otherwise, error.
    // @remark please use simple_dht11_read().
    int sample(int pin, byte data[40]);
    // parse the 40bits data to temperature and humidity.
    // @remark please use simple_dht11_read().
    int parse(byte data[40], byte* ptemperature, byte*
phumidity);
};

#endif
```

LiquidCristal.h

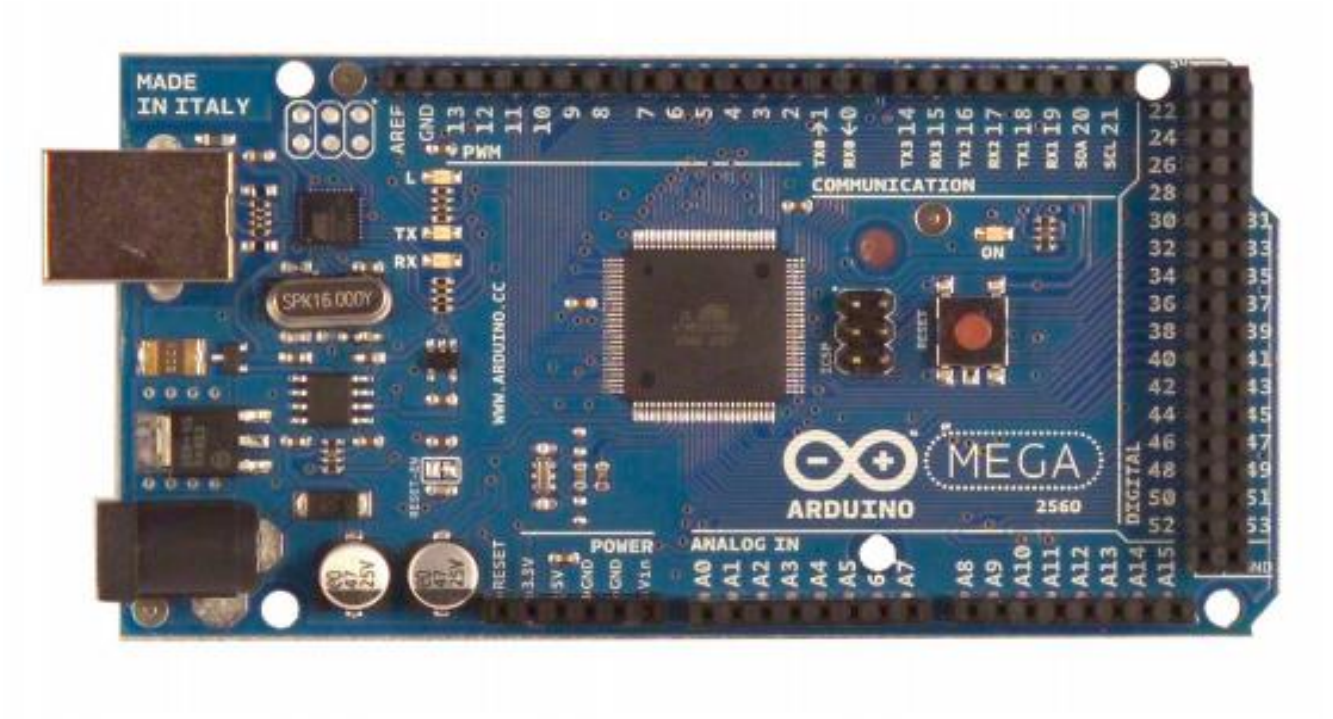
```
#ifndef LiquidCrystal_h
#define LiquidCrystal_h

#include <inttypes.h>
#include "Print.h"

#define LCD_CLEARDISPLAY 0x01
#define LCD_RETURNHOME 0x02
#define LCD_ENTRYMODESET 0x04
#define LCD_DISPLAYCONTROL 0x08
#define LCD_CURSORSHIFT 0x10
#define LCD_FUNCTIONSET 0x20
#define LCD_SETCGRAMADDR 0x40
#define LCD_SETDDRAMADDR 0x80

// flags for display entry mode
#define LCD_ENTRYRIGHT 0x00
#define LCD_ENTRYLEFT 0x02
#define LCD_ENTRYSHIFTINCREMENT 0x01
#define LCD_ENTRYSHIFTDECREMENT 0x00

#define LCD_DISPLAYON 0x04
#define LCD_DISPLAYOFF 0x00
#define LCD_CURSORON 0x02
#define LCD_CURSOROFF 0x00
#define LCD_BLINKON 0x01
#define LCD_BLINKOFF 0x00

#define LCD_DISPLAYMOVE 0x08
#define LCD_CURSORMOVE 0x00
#define LCD_MOVERIGHT 0x04
#define LCD_MOVELEFT 0x00

#define LCD_8BITMODE 0x10
#define LCD_4BITMODE 0x00
#define LCD_2LINE 0x08
#define LCD_1LINE 0x00
#define LCD_5x10DOTS 0x04
#define LCD_5x8DOTS 0x00

class LiquidCrystal : public Print {
public:
  LiquidCrystal(uint8_t rs, uint8_t enable,
              uint8_t d0, uint8_t d1, uint8_t d2, uint8_t d3,
              uint8_t d4, uint8_t d5, uint8_t d6, uint8_t d7);
  LiquidCrystal(uint8_t rs, uint8_t rw, uint8_t enable,
              uint8_t d0, uint8_t d1, uint8_t d2, uint8_t d3,
              uint8_t d4, uint8_t d5, uint8_t d6, uint8_t d7);
  LiquidCrystal(uint8_t rs, uint8_t rw, uint8_t enable,
              uint8_t d0, uint8_t d1, uint8_t d2, uint8_t d3);
  LiquidCrystal(uint8_t rs, uint8_t enable,
              uint8_t d0, uint8_t d1, uint8_t d2, uint8_t d3);
```

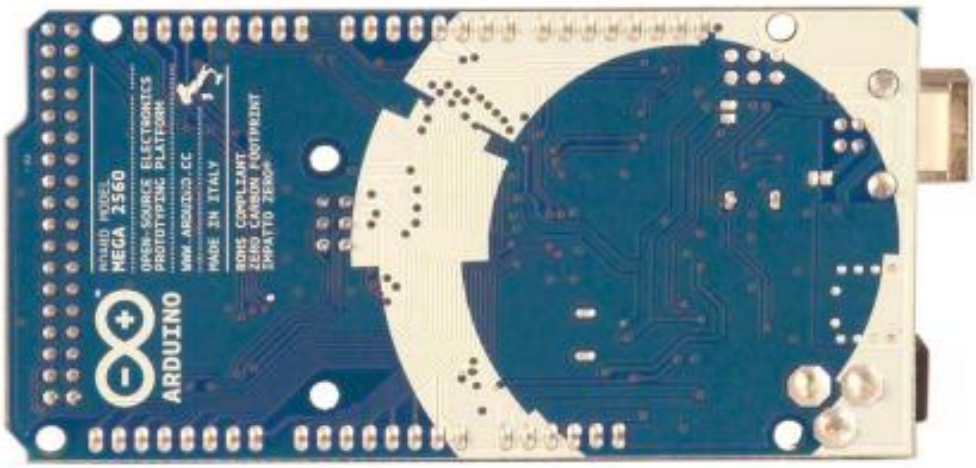
```
  void init(uint8_t fourbitmode, uint8_t rs, uint8_t rw, uint8_t
enable,
          uint8_t d0, uint8_t d1, uint8_t d2, uint8_t d3,
          uint8_t d4, uint8_t d5, uint8_t d6, uint8_t d7);

  void begin(uint8_t cols, uint8_t rows, uint8_t charsize =
LCD_5x8DOTS);

  void clear();
  void home();

  void noDisplay();
  void display();
  void noBlink();
  void blink();
  void noCursor();
  void cursor();
  void scrollDisplayLeft();
  void scrollDisplayRight();
  void leftToRight();
  void rightToLeft();
  void autoscroll();
  void noAutoscroll();

  void setRowOffsets(int row1, int row2, int row3, int row4);
  void createChar(uint8_t, uint8_t[]);
  void setCursor(uint8_t, uint8_t);
  virtual size_t write(uint8_t);
  void command(uint8_t);

  using Print::write;
private:
  void send(uint8_t, uint8_t);
  void write4bits(uint8_t);
  void write8bits(uint8_t);
  void pulseEnable();

  uint8_t _rs_pin; // LOW: command.  HIGH: character.
  uint8_t _rw_pin; // LOW: write to LCD.  HIGH: read from LCD.
  uint8_t _enable_pin; // activated by a HIGH pulse.
  uint8_t _data_pins[8];

  uint8_t _displayfunction;
  uint8_t _displaycontrol;
  uint8_t _displaymode;

  uint8_t _initialized;

  uint8_t _numlines;
  uint8_t _row_offsets[4];
};

#endif
```

Annex III: Datasheets

Arduino Mega 2560

Arduino Mega 2560





The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

Schematic & Reference Design

EAGLE files: arduino-mega2560-reference-design.zip

Schematic: arduino-mega2560-schematic.pdf

Summary

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

Power

The Arduino Mega can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.

- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.

- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

- **GND.** Ground pins.

Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the **EEPROM library**).

Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using **pinMode()**, **digitalWrite()**, and **digitalRead()** functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.

- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the **attachInterrupt()** function for details.

- **PWM: 0 to 13.** Provide 8-bit PWM output with the **analogWrite()** function.

- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication using the **SPI library**. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Uno, Duemilanove and Diecimila.

- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

- **I^2C: 20 (SDA) and 21 (SCL).** Support I^2C (TWI) communication using the **Wire library** (documentation on the Wiring website). Note that these pins are not in the same location as the I^2C pins on the Duemilanove or Diecimila.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analogReference() function.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

Communication

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega8U2 on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Mega2560's digital pins.

The ATmega2560 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation on the Wiring website for details. For SPI communication, use the SPI library.

Programming

The Arduino Mega can be programmed with the Arduino software (download). For details, see the reference and tutorials.

The ATmega2560 on the Arduino Mega comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

The ATmega8U2 firmware source code is available in the Arduino repository. The ATmega8U2 is loaded with a DFU bootloader, which can be activated by connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See this user-contributed tutorial for more information.

Automatic (Software) Reset

Rather then requiring a physical press of the reset button before an upload, the Arduino Mega2560 is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega2560 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can

have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Mega2560 is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Mega2560. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Mega2560 contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

The Arduino Mega2560 has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics and Shield Compatibility

The maximum length and width of the Mega2560 PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

The Mega2560 is designed to be compatible with most shields designed for the Uno, Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega2560 and Duemilanove / Diecimila. ***Please note that I^2C is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Diecimila (analog inputs 4 and 5).***

AX-1838HS

INFRARED RECEIVER MODULE

● Description

TheAX-1838HS is miniaturized infrared receivers for remote control and other applications requiring improved ambient light rejection.

The separate PIN diode and preamplifier IC are assembled on a single leadframe.

The epoxy package contains a special IR filter.

This module has excellent performance even in disturbed ambient light applications and provides protection against uncontrolled output pulses.

● Features

? Photo detector and preamplifier in one package .
? Internal filter for PCM frequency.
? Inner shield,good anti-interference ability.
? High immunity against ambient light.
? Improved shielding against electric field disturbance
? 3.0V or 5.0V supply voltage; low power consumption.
? TTL and CMOS compatibility.
? **8ms data pause time codes are acceptable** .

● Applications:

1. Optical switch
2. Light detecting protion of remote contol
   ? AV instruments such as Audio,TV,VCR,CD,MD,DVD,etc.
   ? Home appliances such as Air-conditioner,Fan,etc.
   ? CATV set top boxes
   ? Multi-media Equipment

● Absolute Maximum Ratings (Ta=25℃)

Parameter	Symbol	Ratings	Unit	Notice
Supply Voltage	Vs	2.1-6.5	V	i
Operating Temperature	Topr	-20~+65	℃	i
Storage Temperature	Tstg	-40~+85	℃	i
Soldering Temperature	Tsd	260	℃	4mm from mold body less than 5 sec

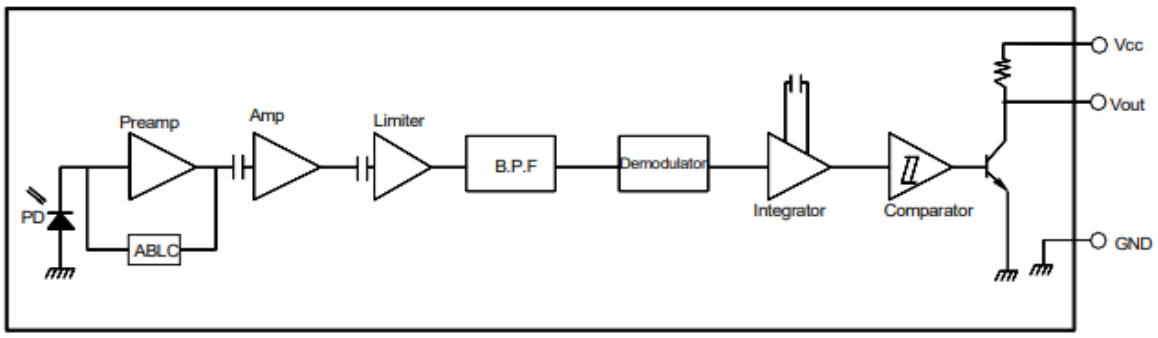
Electrical And Optical Characteristics (Ta=25℃)

Parameter	Symbol	Ratings			Unit	Condition
		Min.	Typ.	Max.		
Supply Voltage	Vs	2.1	-	5.5	V	
Supply Current	Icc	¡	¡	1.5	mA	No signal input
Reception Distance	L_0	17	¡	¡	m	At the ray axis*1
	L_{45}	8	¡	¡		
B.P.F Center Frequency	fo	¡	38	¡	KHz	
Peak Wavelength	λp	¡	940	¡	nm	
Half Angle	θ	¡	45	¡	deg	At the ray axis *1
High Level Pulse Width	T_H	400	¡	800	μS	At the ray axis *2
Low Level Pulse Width	T_L	400	¡	800	μS	
High Level Output Voltage	V_H	4.5	¡	¡	V	
Low Level Output Voltage	V_L	¡	¡	0.5	V	

*1:The ray receiving surface at a vertex and relation to the ray axis in the range of θ=0° and θ=45°

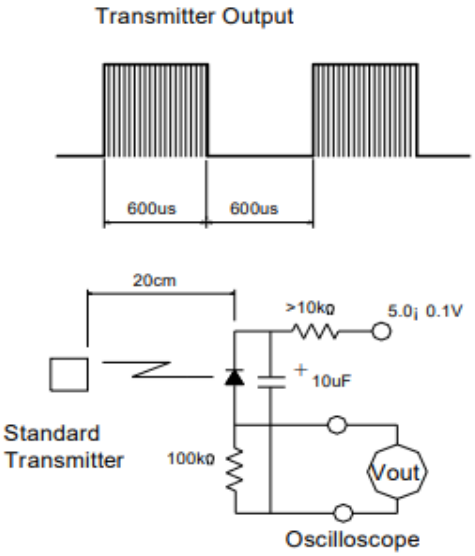
*2:A range from 30cm to the arrival distance. Average value of 50 pulses

● BLOCK DIAGRAM

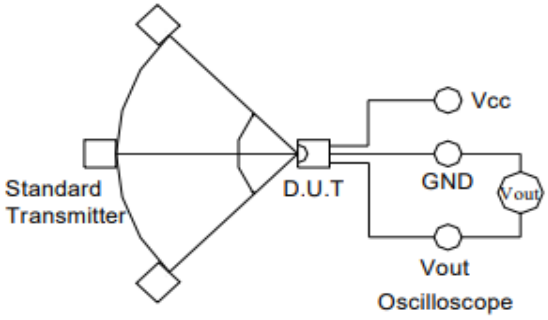


Test Method

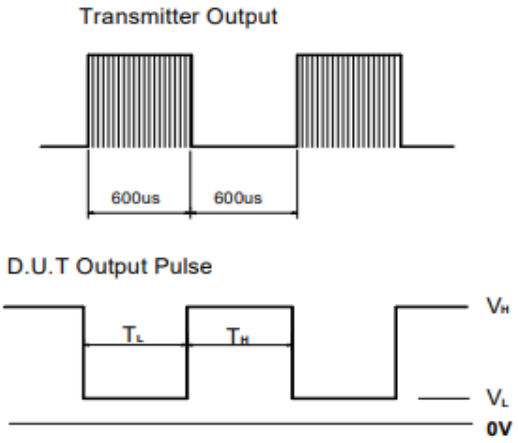
A.Standard Transmitter



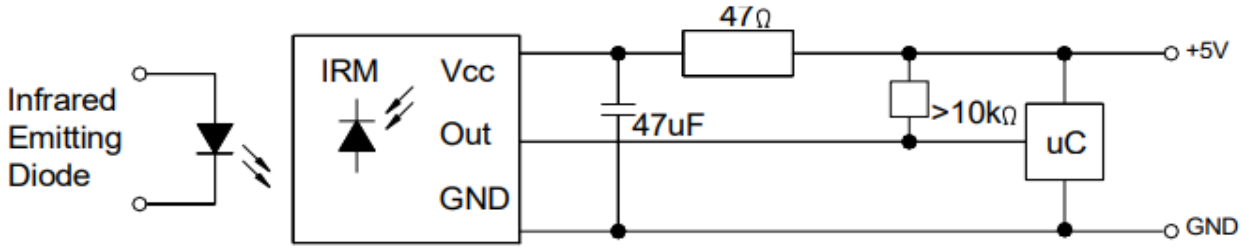
B.Detection Length Test



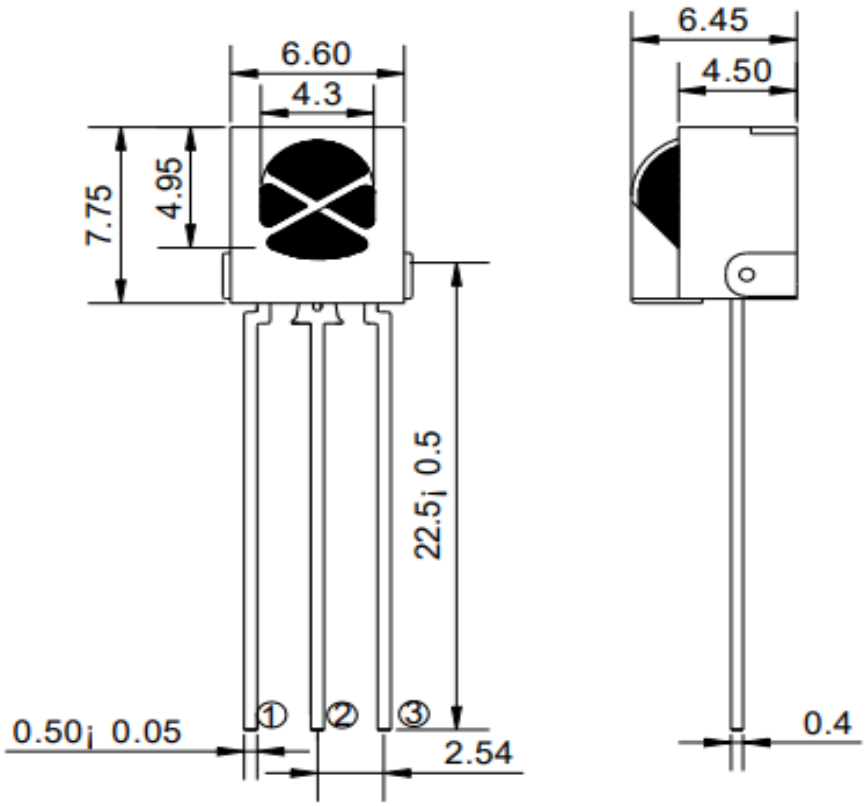
C.Pulse Width Test



Application Circuit



Dimensions:



① OUT
② GND
③ VCC

NOTES:

1.All dimensions are in millimeters (inches).
2.Tolerance is ¡0. 30mm (0.012¡¡) unless otherwise specified.
3.Specifications are subject to change without notice.

Electrical And Optical Curves(Ta=25℃)

Fig.1 Relative Spectral Sensitivity vs. Wavelength

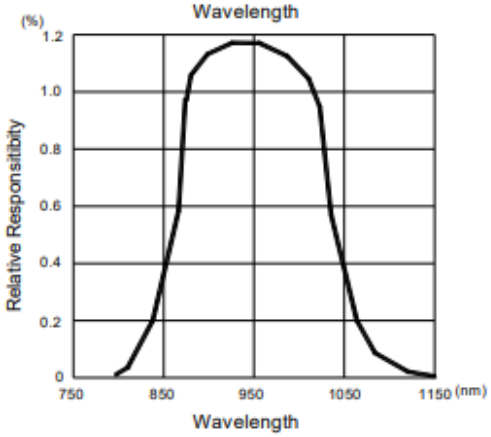


Fig.2 Relative Transmission Distance Vs. Direction

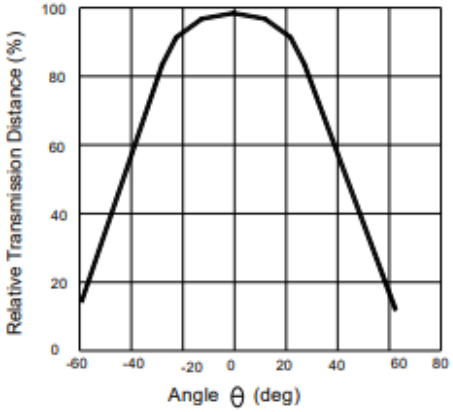


Fig.3 Frequency Dependence of Responsivity

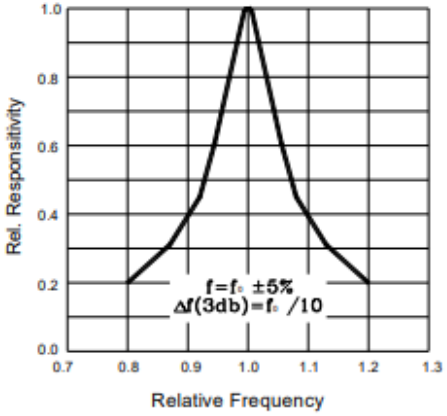


Fig.4 Supply Current vs. Ambient Temperature

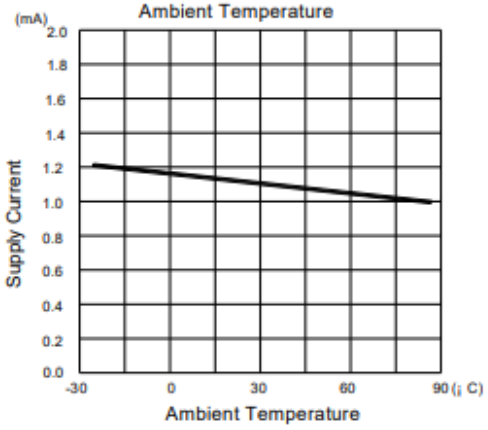
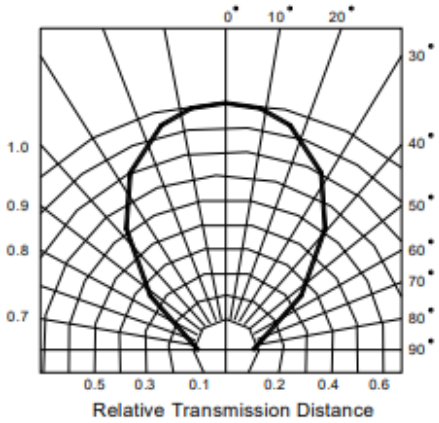


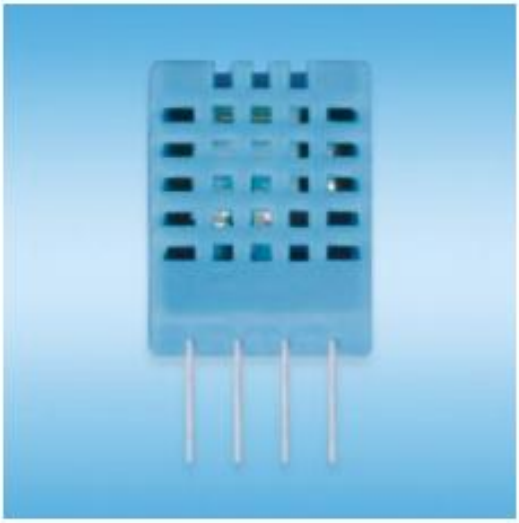
Fig.5 Relative Transmission Distance vs. Direction



DHT11

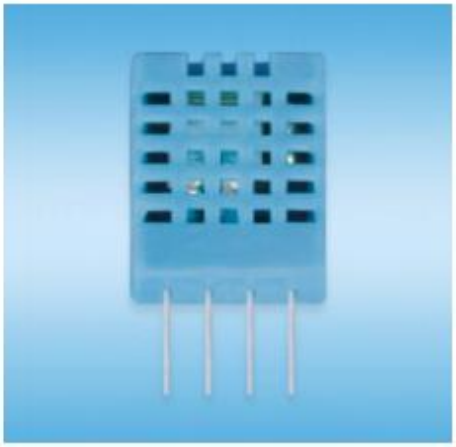
Temperature and humidity module

DHT11 Product Manual



1、Product Overview

DHT11 digital temperature and humidity sensor is a composite Sensor contains a calibrated digital signal output of the temperature and humidity. Application of a dedicated digital modules collection technology and the temperature and humidity sensing technology, to ensure that the product has high reliability and excellent long-term stability. The sensor includes a resistive sense of wet components and an NTC temperature measurement devices, and connected with a high-performance 8-bit microcontroller.



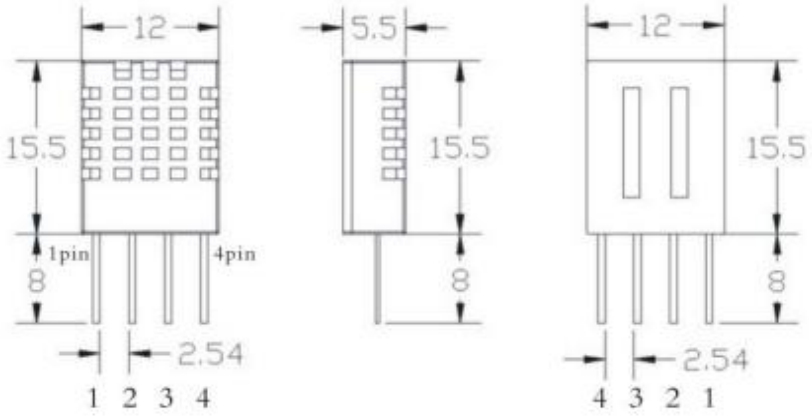
2、Applications

HVAC, dehumidifier, testing and inspection equipment, consumer goods, automotive, automatic control, data loggers, weather stations, home appliances, humidity regulator, medical and other humidity measurement and control.

3、Features

Low cost, long-term stability, relative humidity and temperature measurement, excellent quality, fast response, strong anti-interference ability, long distance signal transmission, digital signal output, and precise calibration.

4、Dimensions (unit: mm)



5、Product parameters

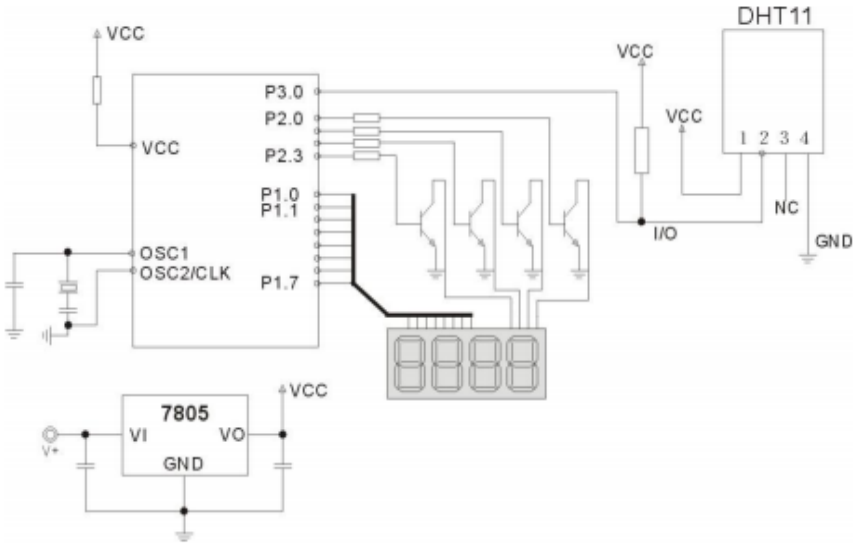
Relative humidity
Resolution: 16Bit
Repeatability: ±1% RH
Accuracy: At 25℃ ±5% RH
Interchangeability: fully interchangeable
Response time: 1 / e (63%) of 25℃ 6s
1m / s air 6s
Hysteresis: <± 0.3% RH
Long-term stability: <± 0.5% RH / yr in

Temperature
Resolution: 16Bit
Repeatability: ±0.2℃
Range: At 25℃ ±2℃
Response time: 1 / e (63%) 10S

Electrical Characteristics
Power supply: DC 3.5 ~ 5.5V
Supply Current: measurement 0.3mA standby 60μ A
Sampling period: more than 2 seconds

Pin Description
1, the VDD power supply 3.5 ~ 5.5V DC
2 DATA serial data, a single bus
3, NC, empty pin
4, GND ground, the negative power

6、Typical circuit



Microprocessor and DHT11 of connection typical application circuit as shown above, DATA pull the microprocessor I / O ports are connected.

1. Typical application circuit recommended in the short cable length of 20 meters on the 5.1K pull–up resistor, the resistance of greater than 20 meters under the pull–up resistor on the lower of the actual situation.

2. When using a 3.5V voltage supply cable length shall not be greater than 20cm. Otherwise, the line voltage drop will cause the sensor power supply shortage, caused by measurement error.

3. Each read out the temperature and humidity values are the results of the last measurement For real–time data, sequential read twice, but is not recommended to repeatedly read the sensors, each read sensor interval is greater than 5 seconds can be obtainedaccurate data.

7、Serial communication instructions (single–wire bi–directional)

◎Single bus Description

DHT11 uses a simplified single–bus communication. Single bus that only one data line, the system of data exchange, control by a single bus to complete. Device (master or slave) through an open–drain or tri–state port connected to the data line to allow the device does not send data to release the bus, while other devices use the bus; single bus usually require an external one about 5.1kΩ pull–up resistor, so that when the bus is idle, its status is high. Because they are the master–slave structure, and only when the host calls the slave, the slave can answer, the host access devices must strictly follow the single–bus sequence, if the chaotic sequence, the device will not respond to the host.

◎Single bus to transfer data defined

DATA For communication and synchronization between the microprocessor and DHT11, single–bus data format, a transmission of 40 data, the high first–out.

Data format:

The 8bit humidity integer data + 8bit the Humidity decimal data +8 bit temperature integer data + 8bit fractional temperature data +8 bit parity bit.

◎Parity bit data definition

"8bit humidity integer data + 8bit humidity decimal data +8 bit temperature integer data + 8bit temperature fractional data" 8bit checksum is equal to the results of the last eight.

Example 1: 40 data is received:

0011 0101	0000 0000	0001 1000	0000 0000	0100 1101
High humidity 8	Low humidity 8	High temp. 8	Low temp. 8	Parity bit

Calculate:

0011 0101+0000 0000+0001 1000+0000 0000= 0100 1101

Received data is correct:

Humidity: 0011 0101=35H=53%RH

Temperature: 0001 1000=18H=24℃

Example 2: 40 data is received:

0011 0101	0000 0000	0001 1000	0000 0000	0100 1001
High humidity 8	Low humidity 8	High temp. 8	Low temp. 8	Parity bit

Calculate:

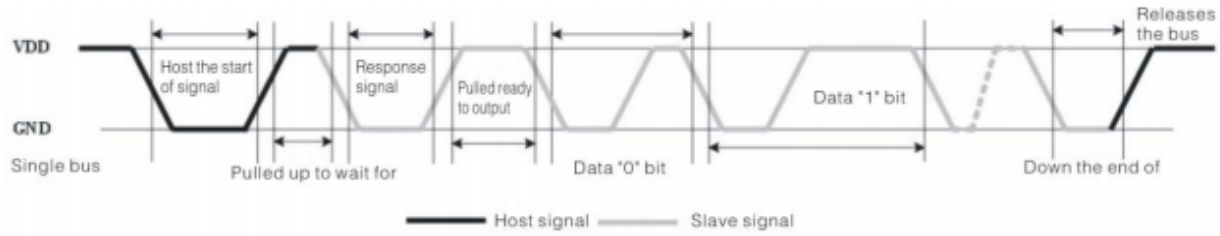
0011 0101+0000 0000+0001 1000+0000 0000 = 0100 1101

01001101≠0100 1001

The received data is not correct, give up, to re-receive data.

◎Data Timing Diagram

User host (MCU) to send a signal, DHT11 converted from low-power mode to high-speed mode, until the host began to signal the end of the DHT11 send a response signal to send 40bit data, and trigger a letter collection. The signal is sent as shown.



Data Timing Diagram

Note: The host reads the temperature and humidity data from DHT11 always the last measured value, such as twice the measured interval of time is very long, continuous read twice to the second value of real-time temperature and humidity values.

◎Peripherals read steps

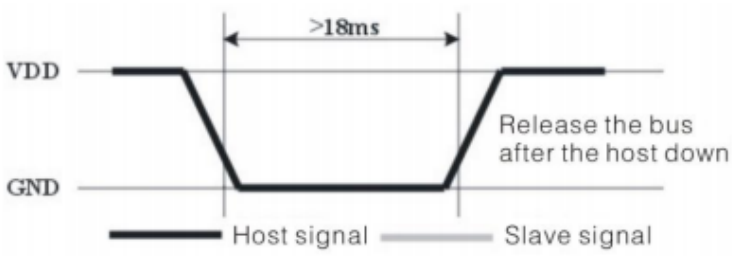
Communication between the master and slave can be done through the following steps (peripherals (such as microprocessors) read DHT11 the data of steps).

Step 1:

After power on DHT11 (DHT11 on after power to wait 1S across the unstable state during this period can not send any instruction), the test environment temperature and humidity data, and record the data, while DHT11 the DATA data lines pulled by pull-up resistor has been to maintainhigh; the DHT11 the DATA pin is in input state, the moment of detection of external signals.

Step 2:

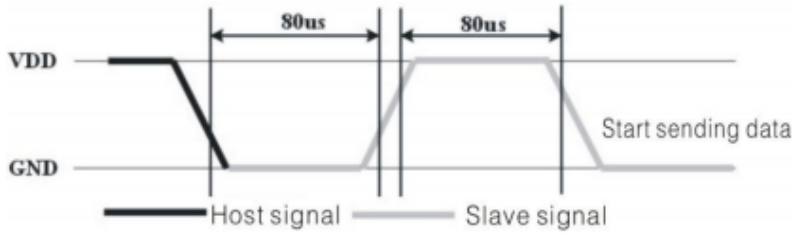
Microprocessor I / O set to output at the same time output low, and low hold time can not be less than 18ms, then the microprocessor I / O is set to input state, due to the pull-up resistor, a microprocessor/ O DHT11 the dATA data lines also will be high, waiting DHT11 to answer signal, send the signal as shown:



Host sends a start signal

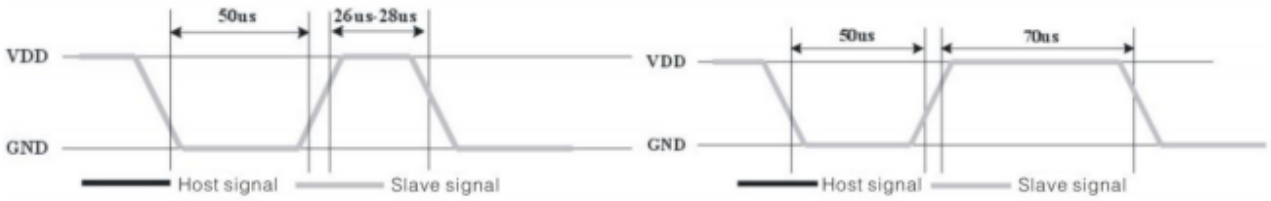
Step 3:

DATA pin is detected to an external signal of DHT11 low, waiting for external signal low end the delay DHT11 DATA pin in the output state, the output low of 80 microseconds as the response signal, followed by the output of 80 micro-seconds of high notification peripheral is ready to receive data, the microprocessor I / O at this time in the input state is detected the I / O low (DHT11 response signal), wait 80 microseconds highdata receiving and sending signals as shown:



Step 4:

Output by DHT11 the DATA pin 40, the microprocessor receives 40 data bits of data "0" format: the low level of 50 microseconds and 26-28 microseconds according to the changes in the I / O levellevel, bit data "1" format: the high level of low plus, 50 microseconds to 70 microseconds. Bit data "0", "1" signal format as shown:



Bit data "0" bit format　　Bit data "1" bit format

End signal:

Continue to output the low 50 microseconds after DHT11 the DATA pin output 40 data, and changed the input state, along with pull-up resistor goes high. But DHT11 internal re-test environmental temperature and humidity data, and record the data, waiting for the arrival of the external signal.

8、Application of information

1. Work and storage conditions

Outside the sensor the proposed scope of work may lead to temporary drift of the signal up to 300%RH. Return to normal working conditions, sensor calibration status will slowly toward recovery. To speed up the recovery process may refer to "resume processing". Prolonged use of non-normal operating conditions, will accelerate the aging of the product.

Avoid placing the components on the long-term condensation and dry environment, as well as the following environment.

A, salt spray

B, acidic or oxidizing gases such as sulfur dioxide, hydrochloric acid

Recommended storage environment

Temperature: 10 ~ 40 ℃ Humidity: 60% RH or less

2. The impact of exposure to chemicals

The capacitive humidity sensor has a layer by chemical vapor interference, the proliferation of chemicals in the sensing layer may lead to drift and decreased sensitivity of the measured values. In a pure environment, contaminants will slowly be released. Resume processing as described below will accelerate this process. The high concentration of chemical pollution (such as ethanol) will lead to the complete damage of the sensitive layer of the sensor.

3. The temperature influence

Relative humidity of the gas to a large extent dependent on temperature. Therefore, in the measurement of humidity, should be to ensure that the work of the humidity sensor at the same temperature. With the release of heat of electronic components share a printed circuit board, the installation should be as far as possible the sensor away from the electronic components and mounted below the heat source, while maintaining good ventilation of the enclosure. To reduce the thermal conductivity sensor and printed circuit board copper plating should be the smallest possible, and leaving a gap between the two.

4. Light impact

Prolonged exposure to sunlight or strong ultraviolet radiation, and degrade performance.

5. Resume processing

Placed under extreme working conditions or chemical vapor sensor, which allows it to return to the status of calibration by the following handler. Maintain two hours in the humidity conditions of 45℃ and <10% RH (dry); followed by 20-30℃ and> 70% RH humidity conditions to maintain more than five hours.

6. Wiring precautions

The quality of the signal wire will affect the quality of the voltage output, it is recommended to use high quality shielded cable.

7. Welding information

Manual welding, in the maximum temperature of 300℃ under the conditions of contact time shall be less than 3 seconds.

8. Product upgrades

Details, please the consultation Aosong electronics department.

9、The license agreement

Without the prior written permission of the copyright holder, shall not in any form or by any means, electronic or mechanical (including photocopying), copy any part of this manual, nor shall its contents be communicated to a third party. The contents are subject to change without notice.

The Company and third parties have ownership of the software, the user may use only signed a contract or software license.

10、Warnings and personal injury

This product is not applied to the safety or emergency stop devices, as well as the failure of the product may result in injury to any other application, unless a particular purpose or use authorized. Installation, handling, use or maintenance of the product refer to product data sheets and application notes. Failure to comply with this recommendation may result in death and serious personal injury. The Company will bear all damages resulting personal injury or death, and waive any claims that the resulting subsidiary company managers and employees and agents, distributors, etc. that may arise, including: a variety of costs, compensation costs, attorneys' fees, and so on.

11、Quality Assurance

The company and its direct purchaser of the product quality guarantee period of three months (from the date of delivery). Publishes the technical specifications of the product data sheet shall prevail. Within the warranty period, the product was confirmed that the quality is really defective, the company will provide free repair or replacement. The user must satisfy the following conditions:

① The product is found defective within 14 days written notice to the Company;
② The product shall be paid by mail back to the company;
③ The product should be within the warranty period.

The Company is only responsible for those used in the occasion of the technical condition of the product defective product. Without any guarantee, warranty or written statement of its products used in special applications. Company for its products applied to the reliability of the product or circuit does not make any commitment.

DS3231

Extremely Accurate I²C-Integrated RTC/TCXO/Crystal

DS3231

General Description

The DS3231 is a low-cost, extremely accurate I²C real-time clock (RTC) with an integrated temperature-compensated crystal oscillator (TCXO) and crystal. The device incorporates a battery input, and maintains accurate timekeeping when main power to the device is interrupted. The integration of the crystal resonator enhances the long-term accuracy of the device as well as reduces the piece-part count in a manufacturing line. The DS3231 is available in commercial and industrial temperature ranges, and is offered in a 16-pin, 300-mil SO package.

The RTC maintains seconds, minutes, hours, day, date, month, and year information. The date at the end of the month is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The clock operates in either the 24-hour or 12-hour format with an AM/PM indicator. Two programmable time-of-day alarms and a programmable square-wave output are provided. Address and data are transferred serially through an I²C bidirectional bus.

A precision temperature-compensated voltage reference and comparator circuit monitors the status of VCC to detect power failures, to provide a reset output, and to automatically switch to the backup supply when necessary. Additionally, the RST pin is monitored as a pushbutton input for generating a reset externally.

Applications

Servers	Utility Power Meters
Telematics	GPS

Pin Configuration appears at end of data sheet.

Features

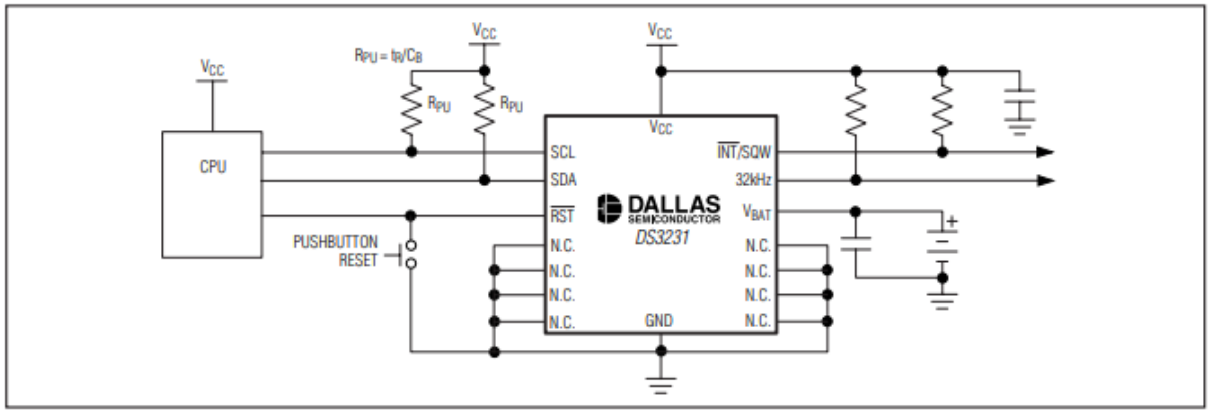
- **Accuracy ±2ppm from 0°C to +40°C**
- **Accuracy ±3.5ppm from -40°C to +85°C**
- **Battery Backup Input for Continuous Timekeeping**
- **Operating Temperature Ranges**
  Commercial: 0°C to +70°C
  Industrial: -40°C to +85°C
- **Low-Power Consumption**
- **Real-Time Clock Counts Seconds, Minutes, Hours, Day, Date, Month, and Year with Leap Year Compensation Valid Up to 2100**
- **Two Time-of-Day Alarms**
- **Programmable Square-Wave Output**
- **Fast (400kHz) I²C Interface**
- **3.3V Operation**
- **Digital Temp Sensor Output: ±3°C Accuracy**
- **Register for Aging Trim**
- **RST Output/Pushbutton Reset Debounce Input**
- **Underwriters Laboratory (UL) Recognized**

Ordering Information

PART	TEMP RANGE	PIN-PACKAGE	TOP MARK
DS3231S	0°C to +70°C	16 SO	DS3231
DS3231SN	-40°C to +85°C	16 SO	DS3231N
DS3231S#	0°C to +70°C	16 SO	DS3231S
DS3231SN#	-40°C to +85°C	16 SO	DS3231SN

 # Denotes a RoHS-compliant device that may include lead that is exempt under RoHS requirements. The lead finish is JESD97 category e3, and is compatible with both lead-based and lead-free soldering processes. A "#" anywhere on the top mark denotes a RoHS-compliant device.

Typical Operating Circuit



Extremely Accurate I²C-Integrated RTC/TCXO/Crystal

ABSOLUTE MAXIMUM RATINGS

Voltage Range on V_{CC}, V_{BAT}, 32kHz, SCL, SDA, $\overline{RST}$, $\overline{INT}$/SQW Relative to Ground ..... -0.3V to +6.0V
Operating Temperature Range (noncondensing) ..... -40°C to +85°C
Junction Temperature ..... +125°C
Storage Temperature Range ..... -40°C to +85°C
Lead Temperature (Soldering, 10s) ..... +260°C/10s
Soldering Temperature ..... See the *Handling, PC Board Layout, and Assembly* section.

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

RECOMMENDED DC OPERATING CONDITIONS

(T_A = T_{MIN} to T_{MAX}, unless otherwise noted.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Supply Voltage	V_{CC}		2.3	3.3	5.5	V
	V_{BAT}		2.3	3.0	5.5	V
Logic 1 Input SDA, SCL	V_{IH}		0.7 x V_{CC}		V_{CC} + 0.3	V
Logic 0 Input SDA, SCL	V_{IL}		-0.3		+0.3 x V_{CC}	V
Pullup Voltage (SDA, SCL, 32kHz, $\overline{INT}$/SQW)	V_{PU}	V_{CC} = 0V			5.5V	V

ELECTRICAL CHARACTERISTICS

(V_{CC} = 2.3V to 5.5V, V_{CC} = Active Supply (see Table 1), T_A = T_{MIN} to T_{MAX}, unless otherwise noted.) (Typical values are at V_{CC} = 3.3V, V_{BAT} = 3.0V, and T_A = +25°C, unless otherwise noted.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Active Supply Current	I_{CCA}	(Notes 3, 4)	V_{CC} = 3.63V			200	µA
			V_{CC} = 5.5V			300	
Standby Supply Current	I_{CCS}	I²C bus inactive, 32kHz output on, SQW output off (Note 4)	V_{CC} = 3.63V			110	µA
			V_{CC} = 5.5V			170	
Temperature Conversion Current	$I_{CCSCONV}$	I²C bus inactive, 32kHz output on, SQW output off	V_{CC} = 3.63V			575	µA
			V_{CC} = 5.5V			650	
Power-Fail Voltage	V_{PF}			2.45	2.575	2.70	V
Logic 0 Output, 32kHz, $\overline{INT}$/SQW, SDA	V_{OL}	I_{OL} = 3mA				0.4	V
Logic 0 Output, $\overline{RST}$	V_{OL}	I_{OL} = 1mA				0.4	V
Output Leakage Current 32kHz, $\overline{INT}$/SQW, SDA	I_{LO}	Output high impedance		-1	0	+1	µA
Input Leakage SCL	I_{LI}			-1		+1	µA
$\overline{RST}$ Pin I/O Leakage	I_{OL}	$\overline{RST}$ high impedance (Note 5)		-200		+10	µA
V_{BAT} Leakage Current (V_{CC} Active)	I_{BATLKG}				25	100	nA

Extremely Accurate I²C-Integrated RTC/TCXO/Crystal

ELECTRICAL CHARACTERISTICS (continued)

(V_{CC} = 2.3V to 5.5V, V_{CC} = Active Supply (see Table 1), T_A = T_{MIN} to T_{MAX}, unless otherwise noted.) (Typical values are at V_{CC} = 3.3V, V_{BAT} = 3.0V, and T_A = +25°C, unless otherwise noted.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Output Frequency	f_{OUT}	V_{CC} = 3.3V or V_{BAT} = 3.3V			32.768		kHz
Frequency Stability vs. Temperature (Commercial)	$\Delta f/f_{OUT}$	V_{CC} = 3.3V or V_{BAT} = 3.3V, aging offset = 00h	0°C to +40°C			±2	ppm
			>40°C to +70°C			±3.5	
Frequency Stability vs. Temperature (Industrial)	$\Delta f/f_{OUT}$	V_{CC} = 3.3V or V_{BAT} = 3.3V, aging offset = 00h	-40°C to <0°C			±3.5	ppm
			0°C to +40°C			±2	
			>40°C to +85°C			±3.5	
Frequency Stability vs. Voltage	$\Delta f/V$				1		ppm/V
Trim Register Frequency Sensitivity per LSB	Δf/LSB	Specified at:	-40°C		0.7		ppm
			+25°C		0.1		
			+70°C		0.4		
			+85°C		0.8		
Temperature Accuracy	Temp	V_{CC} = 3.3V or V_{BAT} = 3.3V		-3		+3	°C
Crystal Aging	$\Delta f/f_0$	After reflow, not production tested	First year		±1.0		ppm
			0–10 years		±5.0		

ELECTRICAL CHARACTERISTICS

(**V_{CC} = 0V, V_{BAT} = 2.3V to 5.5V**, T_A = T_{MIN} to T_{MAX}, unless otherw.ise noted.) (Note 1)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Active Battery Current	I_{BATA}	$\overline{EOSC}$ = 0, BBSQW = 0, SCL = 400kHz (Note 4)	V_{BAT} = 3.63V			70	µA
			V_{BAT} = 5.5V			150	
Timekeeping Battery Current	I_{BATT}	$\overline{EOSC}$ = 0, BBSQW = 0, EN32kHz = 1, SCL = SDA = 0V or SCL = SDA = V_{BAT} (Note 4)	V_{BAT} = 3.63V		0.84	3.0	µA
			V_{BAT} = 5.5V		1.0	3.5	
Temperature Conversion Current	I_{BATTC}	$\overline{EOSC}$ = 0, BBSQW = 0, SCL = SDA = 0V or SCL = SDA = V_{BAT}	V_{BAT} = 3.63V			575	µA
			V_{BAT} = 5.5V			650	
Data-Retention Current	I_{BATTDR}	$\overline{EOSC}$ = 1, SCL = SDA = 0V, +25°C				100	nA

Extremely Accurate I²C-Integrated RTC/TCXO/Crystal

AC ELECTRICAL CHARACTERISTICS

(V_{CC} = $V_{CC(MIN)}$ to $V_{CC(MAX)}$ or V_{BAT} = $V_{BAT(MIN)}$ to $V_{BAT(MAX)}$, V_{BAT} > V_{CC}, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.) (Note 1)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
SCL Clock Frequency	f_{SCL}	Fast mode	100		400	kHz
		Standard mode	0		100	
Bus Free Time Between STOP and START Conditions	t_{BUF}	Fast mode	1.3			µs
		Standard mode	4.7			
Hold Time (Repeated) START Condition (Note 6)	$t_{HD:STA}$	Fast mode	0.6			µs
		Standard mode	4.0			
Low Period of SCL Clock	t_{LOW}	Fast mode	1.3			µs
		Standard mode	4.7			
High Period of SCL Clock	t_{HIGH}	Fast mode	0.6			µs
		Standard mode	4.0			
Data Hold Time (Notes 7, 8)	$t_{HD:DAT}$	Fast mode	0		0.9	µs
		Standard mode	0		0.9	
Data Setup Time (Note 9)	$t_{SU:DAT}$	Fast mode	100			ns
		Standard mode	250			
Start Setup Time	$t_{SU:STA}$	Fast mode	0.6			µs
		Standard mode	4.7			
Rise Time of Both SDA and SCL Signals (Note 10)	t_R	Fast mode	20 + 0.1C_B		300	ns
		Standard mode			1000	
Fall Time of Both SDA and SCL Signals (Note 10)	t_F	Fast mode	20 + 0.1C_B		300	ns
		Standard mode			300	
Setup Time for STOP Condition	$t_{SU:STO}$	Fast mode	0.6			µs
		Standard mode	4.7			
Capacitive Load for Each Bus Line (Note 10)	C_B				400	pF
Capacitance for SDA, SCL	$C_{I/O}$			10		pF
Pulse Width of Spikes That Must Be Suppressed by the Input Filter	t_{SP}			30		ns
Pushbutton Debounce	PB_{DB}			250		ms
Reset Active Time	t_{RST}			250		ms
Oscillator Stop Flag (OSF) Delay	t_{OSF}	(Note 11)		100		ms
Temperature Conversion Time	t_{CONV}			125	200	ms

POWER-SWITCH CHARACTERISTICS

(T_A = T_{MIN} to T_{MAX})

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
V_{CC} Fall Time; $V_{PF(MAX)}$ to $V_{PF(MIN)}$	t_{VCCF}		300			µs
V_{CC} Rise Time; $V_{PF(MIN)}$ to $V_{PF(MAX)}$	t_{VCCR}		0			µs
Recovery at Power-Up	t_{REC}	(Note 12)		250	300	ms

HL-54S

松乐继电器 ® SONGLE RELAY	RELAY ISO9002	SRD

1. MAIN FEATURES

- Switching capacity available by 10A in spite of small size design for highdensity P.C. board mounting technique.
- UL,CUL,TUV recognized.
- Selection of plastic material for high temperature and better chemical solution performance.
- Sealed types available.
- Simple relay magnetic circuit to meet low cost of mass production.

2. APPLICATIONS

- Domestic appliance, office machine, audio, equipment, automobile, etc.
  (Remote control TV receiver, monitor display, audio equipment high rushing current use application.)

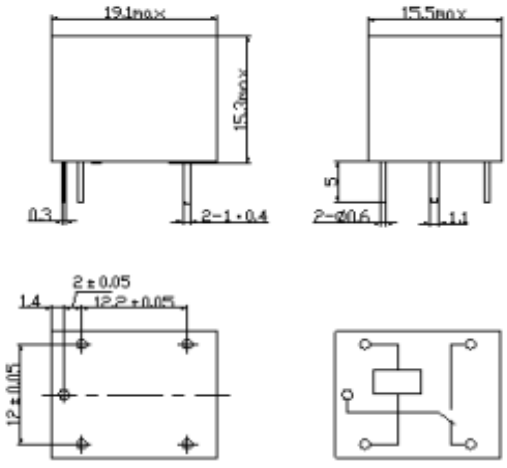
3. ORDERING INFORMATION

SRD	XX VDC	S	L	C
Model of relay	Nominal coil voltage	Structure	Coil sensitivity	Contact form
SRD	03、05、06、09、12、24、48VDC	S:Sealed type	L:0.36W	A:1 form A
				B:1 form B
		F:Flux free type	D:0.45W	C:1 form C

4. RATING

CCC	FILE NUMBER:CH0052885-2000	7A/240VDC
CCC	FILE NUMBER:CH0036746-99	10A/250VDC
UL /CUL	FILE NUMBER: E167996	10A/125VAC 28VDC
TUV	FILE NUMBER: R9933789	10A/240VAC 28VDC

5. DIMENSION(unit:mm) DRILLING(unit:mm) WIRING DIAGRAM



6. COIL DATA CHART (AT20°C)

Coil Sensitivity	Coil Voltage Code	Nominal Voltage (VDC)	Nominal Current (mA)	Coil Resistance (Ω) ±10%	Power Consumption (W)	Pull-In Voltage (VDC)	Drop-Out Voltage (VDC)	Max-Allowable Voltage (VDC)
SRD (High Sensitivity)	03	03	120	25	abt. 0.36W	75%Max.	10% Min.	120%
	05	05	71.4	70				
	06	06	60	100				
	09	09	40	225				
	12	12	30	400				
	24	24	15	1600				
	48	48	7.5	6400				
SRD (Standard)	03	03	150	20	abt. 0.45W	75% Max.	10% Min.	110%
	05	05	89.3	55				
	06	06	75	80				
	09	09	50	180				
	12	12	37.5	320				
	24	24	18.7	1280				
	48	48	10	4500	abt. 0.51W			

7. CONTACT RATING

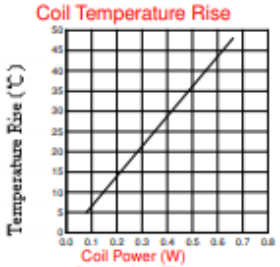
Item \ Type	SRD FORM C	SRD FORM A
Contact Capacity Resistive Load (cosΦ=1)	7A 28VDC 10A 125VAC 7A 240VAC	10A 28VDC 10A 240VAC
Inductive Load (cosΦ=0.4 L/R=7msec)	3A 120VAC 3A 28VDC	5A 120VAC 5A 28VDC
Max. Allowable Voltage	250VAC/110VDC	250VAC/110VDC
Max. Allowable Power Force	800VAC/240W	1200VA/300W
Contact Material	AgCdO	AgCdO

8. PERFORMANCE (at initial value)

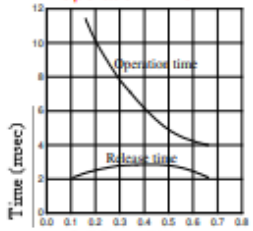
Item \ Type	SRD
Contact Resistance	100mΩ Max.
Operation Time	10msec Max.
Release Time	5msec Max.
Dielectric Strength Between coil & contact Between contacts	 1500VAC 50/60HZ (1 minute) 1000VAC 50/60HZ (1 minute)
Insulation Resistance	100 MΩ Min. (500VDC)
Max. ON/OFF Switching Mechanically Electrically	 300 operation/min 30 operation/min
Ambient Temperature	-25°C to +70°C
Operating Humidity	45 to 85% RH
Vibration Endurance Error Operation	 10 to 55Hz Double Amplitude 1.5mm 10 to 55Hz Double Amplitude 1.5mm
Shock Endurance Error Operation	 100G Min. 10G Min.
Life Expectancy Mechanically Electrically	 10^7 operations. Min. (no load) 10^5 operations. Min. (at rated coil voltage)
Weight	abt. 10grs.

9.REFERENCE DATA

Coil Temperature Rise

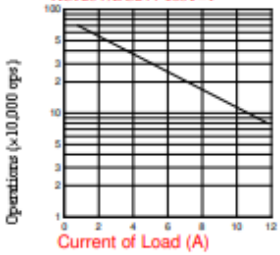


Operation Time

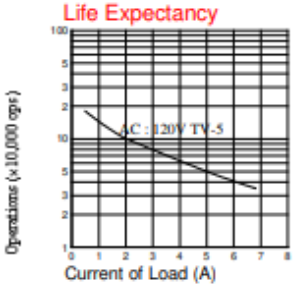


Life Expectancy

AC120V/DC24V cosΦ=1

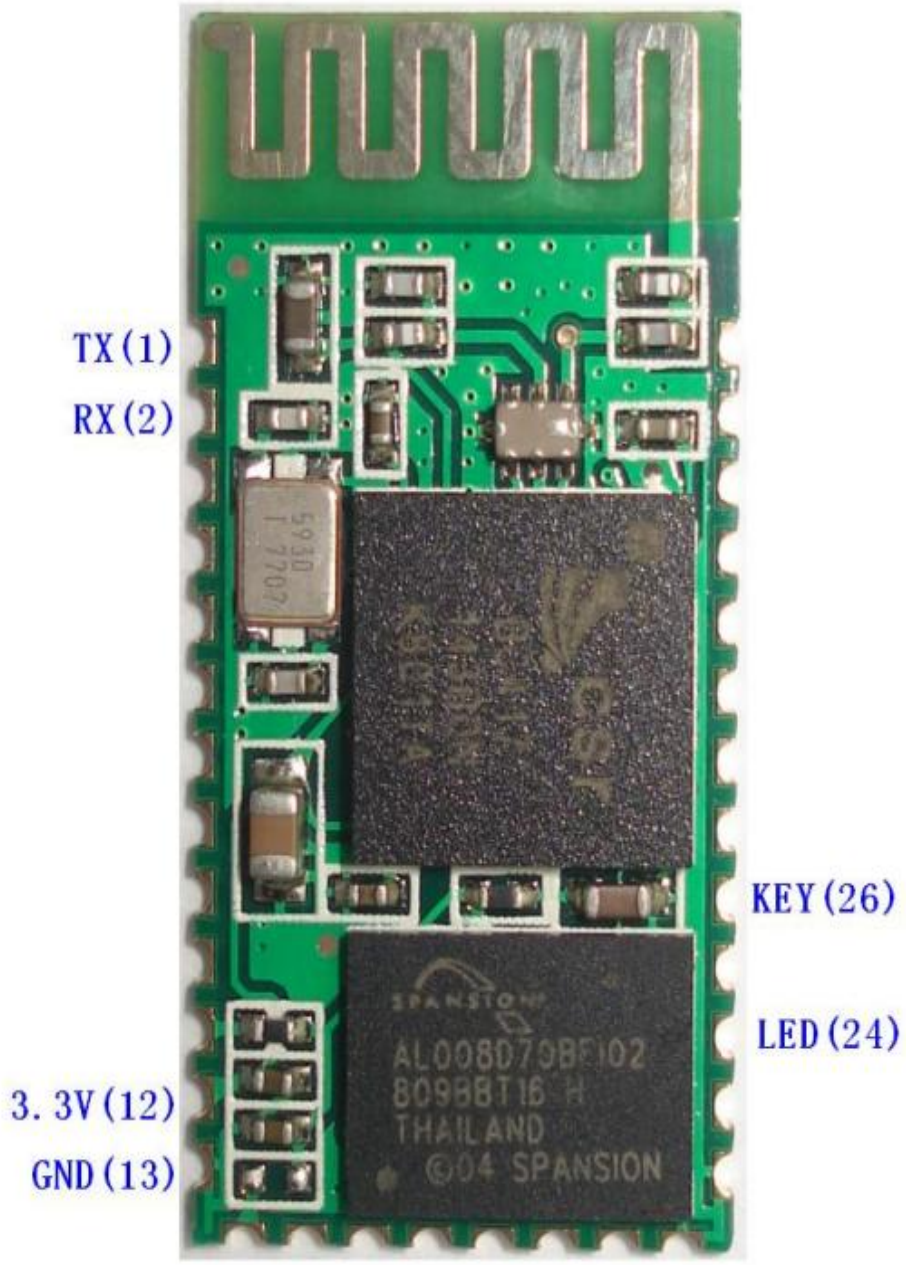


Life Expectancy



HC-06

1. Product's picture



2. Feature

- Wireless transceiver
  - Sensitivity (Bit error rate) can reach -80dBm.
  - The change range of output's power: -4 - +6dBm.
- Function description (perfect Bluetooth solution)
  - Has an EDR module; and the change range of modulation depth: 2Mbps - 3Mbps.
  - Has a build-in 2.4GHz antenna; user needn't test antenna.
  - Has the external 8Mbit FLASH
  - Can work at the low voltage (3.1V~4.2V). The current in pairing is in the range of 30~40mA. The current in communication is 8mA.
  - Standard HCI Port (UART or USB)
  - USB Protocol: Full Speed USB1.1, Compliant With 2.0
  - This module can be used in the SMD.
  - It's made through RoHS process.
  - The board PIN is half hole size.
  - Has a 2.4GHz digital wireless transceiver.
  - Bases at CSR BC04 Bluetooth technology.
  - Has the function of adaptive frequency hopping.
  - Small (27mm×13mm×2mm)
  - Peripherals circuit is simple.
  - It's at the Bluetooth class 2 power level.
  - Storage temperature range: -40 ℃ - 85℃， work temperature range: -25 ℃ - +75℃
  - Any wave inter Interference: 2.4MHz， the power of emitting: 3 dBm.
  - Bit error rate: 0. Only the signal decays at the transmission link, bit error may be produced. For example, when RS232 or TTL is being processed, some signals may decay.

- Low power consumption
- Has high-performance wireless transceiver system
- Low Cost

- Application fields:
  - Bluetooth Car Handsfree Device
  - Bluetooth GPS
  - Bluetooth PCMCIA , USB Dongle
  - Bluetooth Data Transfer
- Software
  - CSR

3. PINs description

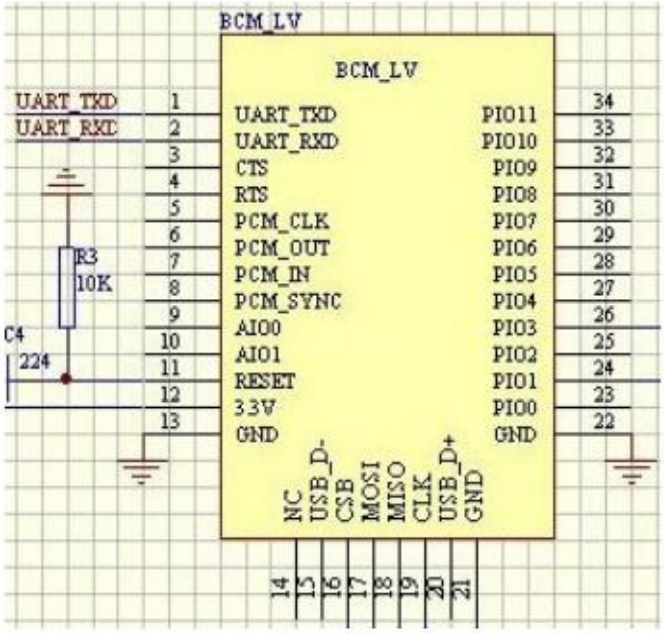


Figure 3 PIN configuration

The PINs at this block diagram is as same as the physical one.

PIN Name	PIN #	Pad type	Description	Note
GND	13 21 22	VSS	Ground pot	
1V8	14	VDD	Integrated 1.8V (+) supply with On-chip linear regulator output within 1.7-1.9V	
VCC	12	3.3V		
AIO0	9	Bi-Directional	Programmable input/output line	
AIO1	10	Bi-Directional	Programmable input/output line	

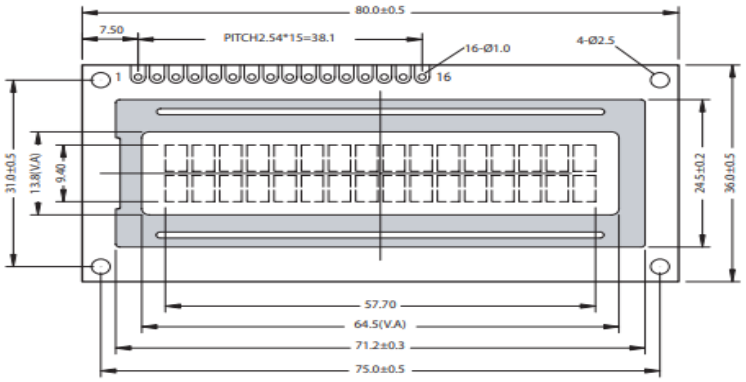
PIO0	23	Bi-Directional RX EN	Programmable input/output line, control output for LNA(if fitted)	
PIO1	24	Bi-Directional TX EN	Programmable input/output line, control output for PA(if fitted)	
PIO2	25	Bi-Directional	Programmable input/output line	
PIO3	26	Bi-Directional	Programmable input/output line	
PIO4	27	Bi-Directional	Programmable input/output line	
PIO5	28	Bi-Directional	Programmable input/output line	
PIO6	29	Bi-Directional	Programmable input/output line	CLK_REQ
PIO7	30	Bi-Directional	Programmable input/output line	CLK_OUT
PIO8	31	Bi-Directional	Programmable input/output line	
PIO9	32	Bi-Directional	Programmable input/output line	
PIO10	33	Bi-Directional	Programmable input/output line	
PIO11	34	Bi-Directional	Programmable input/output line	
RESETB	11	CMOS Input with weak intemal pull-down		
UART_RTS	4	CMOS output, tri-stable with weak internal pull-up	UART request to send, active low	
UART_CTS	3	CMOS input with weak internal pull-down	UART clear to send, active low	
UART_RX	2	CMOS input with weak internal pull-down	UART Data input	
UART_TX	1	CMOS output, Tri-stable with weak internal pull-up	UART Data output	
SPI_MOSI	17	CMOS input with weak internal pull-down	Serial peripheral interface data input	
SPI_CSB	16	CMOS input with weak internal	Chip select for serial peripheral interface, active low	

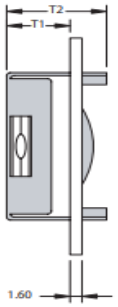
		pull-up		
SPI_CLK	19	CMOS input with weak internal pull-down	Serial peripheral interface clock	
SPI_MISO	18	CMOS input with weak internal pull-down	Serial peripheral interface data Output	
USB_-	15	Bi-Directional		
USB_+	20	Bi-Directional		
1.8V	14		1.8V external power supply input	Default : 1.8V internal powe r supply.
PCM_CLK	5	Bi-Directional		
PCM_OUT	6	CMOS output		
PCM_IN	7	CMOS Input		
PCM_SYNC	8	Bi-Directional		

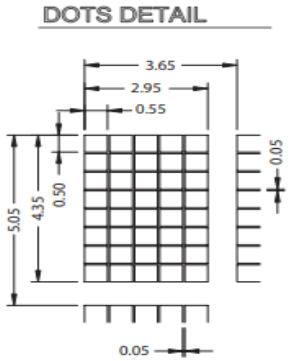
LCD1602A

LONGTECH DISPLAY

LCM1602A







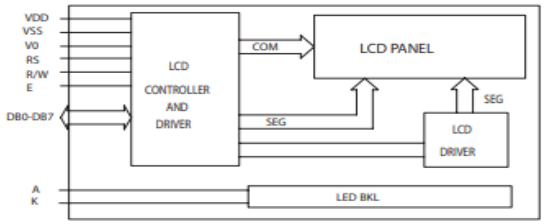
Pin connections

PIN NO	Symbol
1	VSS
2	VDD
3	V0
4	RS
5	R/W
6	E
7	DB0
8	DB1
9	DB2
10	DB3
11	DB4
12	DB5
13	DB6
14	DB7
15	A
16	K

Feature

1. 2-line x16 characters display; 5x8 dots with cursor
2. Built-in controller (KS0066U or Equivalent)
3. +5V power supply(Also available for +3.3V)
4. 1/16 duty cycle;1/5bias;
5. TN, HTN, STN or FSTN mode
6. BKL to be driven by pin1,pin2,or pin15,pin16,or A,K

Block diagram:



Display character address code

Display position	1	2	3	4	5	---	---	---	---	14	15	16
DDRAM address	00	01	02	---	---	---	---	---	---	0EH	0FH	
DDRAM address	40	41	42	---	---	---	---	---	---	4EH	4FH	

Mechanical thinkness

LCD module thinkness	T1	T2	UNIT
Without or EL backlight	4.5	9.5	mm
With edge LED backlight	7.0	11.0	mm
With LED bottom backlight	8.5	13.5	mm

Mechanical data

Item	Standard	Unit
Module dimension	80.0x36.0	mm
Viewing area	64.5x13.8	mm
Dot size	0.55x0.50	mm
Character size	2.95x5.05	mm

Absolute Maximum Rating

Item	Symbol	Standard			Unit
		Min	Typ	Max	
Input voltage	VI	-0.3	--------	5.5	V
Power supply	VDD-VSS	-0.3	--------	VDD	

Electronical characteristics

Item	Symbol	Condition	Standard			Unit
			Min	Typ	Max	
Input voltage	VDD	+5V	4.7	5.0	5.5	V
		+3.3V	2.7	3.3	5.3	V
Supply current	IDD	VDD=5V	------	1.5	4	mA
Recommended LCD driving voltage for normal temp version module	VDD-V0	-20°C	4.8	5.0	5.3	V
		0 °C	4.6	4.8	5.0	
		25°C	4.3	4.5	4.7	
		50°C	4.2	4.3	4.5	
		70°C	4.1	4.2	4.3	
LED forward voltage	VF	25°C	------	4.2	5..0	V
LED forward current	IF	25°C	------	110	-----	mA
EL power supply current	IEL	VEL=110V AC 400Hz	-----	-------	------	mA

Annex IV: Esquema de connexió